(2)

AL-TP-1992-0055

AD-A260 125

ARMSTRONG LABORATORY

STATISTICAL NEURAL NETWORK ANALYSIS PACKAGE (SNNAP) OVERVIEW AND DEMONSTRATION OF FACILITIES

Vince L. Wiggins

RRC, Incorporated
3833 Texas Avenue, Suite 256
Bryan, TX 77802

Kevin M. Borden

Metrica, Incorporated
3833 Texas Avenue, Suite 207
Bryan, TX 77802

Sheree K. Engquist, Major, USAF

HUMAN RESOURCES DIRECTORATE
MANPOWER AND PERSONNEL RESEARCH DIVISION
Brooks Air Force Base, TX 78235-5352

December 1992

Final Technical Paper for Period December 1990 - May 1992

Approved for public release; distribution is unlimited.

AIR FORCE MATERIEL COMMAND
BROOKS AIR FORCE BASE, TEXAS 78235-5000

NOTICES

This paper is published as received and has not been edited by the technical editing staff of the Armstrong Laboratory.

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the Government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

The Office of Public Affairs has reviewed this paper, and it is releasable to the National Technical Information Service, where it will be available to the general public, including foreign nationals.

This paper has been reviewed and is approved for publication.

SHEREE K. ENGQUIST, Major, USAF
Project Scientist

WILLIAM E. ALLEY, Technical Director
Manpower and Personnel Research Division

ROGER W. ALFORD, Lt Col, USAF
Chief, Manpower and Personnel Research Division

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)

2. REPORT DATE
December 1992

3. REPORT TYPE AND DATES COVERED
Final – December 1990 – May 1992

4. TITLE AND SUBTITLE
Statistical Neural Network Analysis Package (SNNAP) Overview and Demonstration of Facilities

5. FUNDING NUMBERS
C – F41689-88-D-0251
PE – 62205F
PR – 7719
TA – 20
WU – 20

6. AUTHOR(S)
Vince L. Wiggins
Kevin M. Borden
Sheree K. Engquist

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
RRC, Incorporated
3833 Texas Avenue, Suite 256
Bryan, TX 77802

Metrica, Incorporated
3833 Texas Avenue, Suite 207
Bryan, TX 77802

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAMES(S) AND ADDRESS(ES)
Armstrong Laboratory
Human Resources Directorate
Manpower and Personnel Research Division
Brooks Air Force Base, TX 78235-5352

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AL-TP-1992-0055

11. SUPPLEMENTARY NOTES
Armstrong Laboratory Technical Monitor: Major Sheree K. Engquist, (210) 536-2257

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

The Statistical Neural Network Analysis Package (SNNAP) was developed to support research and application of neural network personnel models within the Armstrong Laboratory and other government agencies. The package provides extensive facilities for developing and analyzing networks. It utilizes training heuristics developed in prior research to improve out-of-sample performance of network models. SNNAP provides extensive tools for analyzing and visualizing the structure of trained network models. The report provides an overview of SNNAP facilities and an extensive example using SNNAP to analyze the linkage between task performance, task experience, and airman aptitude.

14. SUBJECT TERMS
Back propagation
Computer software package
Neural networks
Personnel system modeling

15. NUMBER OF PAGES
68

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102

CONTENTS

CONTENTS (Continued)

LIST OF FIGURES

List of Figures (Continued)

LIST OF TABLES

PREFACE

This research and development effort was conducted as Task 50, under contract F41689-88-D-0251. This research supports work unit 77192020, Economic Models for Force Management and Costing. This report documents continuing efforts by the Human Resources Directorate of the Armstrong Laboratory to utilize neural networks in the development of personnel models. Prior research on several of the methods used in this report is documented in Wiggins, Looper, & Engquist (1991) and Wiggins, Engquist, & Looper (1992). The focus of these efforts has been to develop tools which provide for a richer and more robust predictive personnel modeling capability.

STATISTICAL NEURAL NETWORK ANALYSIS PACKAGE (SNNAP) OVERVIEW AND DEMONSTRATION OF FACILITIES

SUMMARY

The Statistical Neural Network Analysis Package (SNNAP) is a software environment for developing and analyzing neural network models of decisions, time-series phenomenon, system control, and other input-output relationships. The basic facilities available in SNNAP are documented in this report and a detailed example of analyzing data with SNNAP is reported. SNNAP operates under the Microsoft Windows 3.0 or 3.1 platform and takes complete advantage of the user interface and graphics capabilities of the environment. The package implements training heuristics, developed in prior research, which significantly improve the performance of neural networks in personnel analysis.

The package can utilize three very different neural network architectures: back propagation, probabilistic neural network, and learning vector quantization. Each of these architectures has demonstrated empirical success in several non-personnel areas and back propagation has proven particularly successful in early personnel research. Each of the architectures is based on a different method of developing relationships: back propagation uses layered nonlinear functions, probabilistic neural networks use local kernel based techniques, and learning vector quantization uses a form of basis functions. For any of the architectures, SNNAP contains an expert system which will "suggest" a specific structure and set of parameters for any particular model. This suggestion is based on information provided by the user on the data set to be analyzed.

Another unique aspect of SNNAP is its extensive tools for analyzing and visualizing the response surface of neural network models. Because neural networks can develop complex nonlinear models, understanding the relationships in the model can be difficult. SNNAP contains facilities to provide 3-dimensional views of model response as well as extensions to view relations directly or in the form of impact charts or tables. The software also includes a facility for automatically scanning a model's response surface to identify interesting features in the underlying model.

In an analysis of task performance using a single task from the Precision Measuring Equipment Specialist specialty (324X0), several potentially important relationships were developed by the network model. In particular, the relation between task experience and task performance (as measured by the proportion of steps correctly completed) was found to be highly nonlinear. Early hands-on training was found to be highly indicative of improved task performance, particularly for airmen with low mechanical aptitude. The structure of the linkage between aptitude, experience, and task performance as modeled by the networks could have significant implications on training and selection policy. However, much more extensive modeling across all tasks and more specialties is required.

INTRODUCTION

This task focuses on developing a neural network software system which implements concepts evolved in prior Armstrong Laboratory research. While neural networks are sometimes used for optimization problems, the current system emphasizes the ability of neural networks to identify relationships between inputs from samples of system or individual behavior. In this sense, the networks are used for problems typically approached with statistics, econometrics, clustering, and pattern recognition techniques. The major advantage which neural networks bring to these problems is the ability to extract nonlinear relations and interactions among inputs without prior knowledge of specific functional forms. As demonstrated in prior research (Wiggins, Looper, & Engquist, 1991; Wiggins, Engquist, & Looper, 1992), this ability has allowed the networks to surpass the performance of some established personnel models developed with more traditional techniques.

The Statistical Neural Network Analysis Package (SNNAP) has been developed to operationalize the results of the prior research. It makes available a facility for the training and analysis of neural networks. In particular, two areas which are germane to personnel research, but not completely available in commercial neural network packages, are addressed by SNNAP. The first area is network generalization, or the ability of the network to perform well out-of-sample on data not available during training. With high levels of stochastic error, typical of personnel data, neural networks have a tendency to over-fit and perform poorly out-of-sample. SNNAP includes training heuristics from prior research to significantly improve generalization. The second area is network analysis, or the ability to illustrate the relationships "discovered" by a neural network. Because networks are not constrained to a specific functional form, it is critical to visualize the relationships which networks develop between their inputs (independent variables) and outputs (dependent variables). In addition, SNNAP eases the use of neural networks by including a system to suggest network parameters based on the data being analyzed.

SNNAP is written in Borland C++ using object oriented design concepts to facilitate any future expansion of analysis capabilities or network architectures. It operates in the Windows 3.0 or 3.1 environment and makes complete use of the graphical capabilities of Windows. Information and graphics may be transferred from SNNAP to other windows products using the clipboard.*

This report serves primarily to document the specific facilities implemented in SNNAP. Further documentation of specific neural network architectures and training methods can be found in the appropriate references. The report is organized in two major sections: an overview of SNNAP facilities and a detailed example applying SNNAP to an actual personnel issue. The overview describes all of the major SNNAP facilities including: neural network architectures, data handling, automatic parameter selection, network analysis, and automated surface analysis. In the example, SNNAP is applied to the problem of linking job performance to measurable aptitude and job experience. This example emphasizes SNNAP facilities rather than the theoretical, data, or institutional issues. Data formats for SNNAP are covered in Appendix A and a flowchart for the SNNAP software can be found in Appendix B.

OVERVIEW OF SNNAP FACILITIES

Network Architectures

The heart of any neural network package is the network architectures which it supports. Neural networks are not a single technique, but a rapidly expanding field which has drawn from statistics, pattern recognition, neurobiology, statistical mechanics, and other fields. SNNAP implements three radically different network architectures, each of which has been successful in solving classification and continuous modeling problems. SNNAP allows several networks to be analyzed simultaneously. These networks can be selected to have similar architectures but different parameters or can be selected from different architectures.

In the sections that follow, each of the network architectures will be discussed briefly. This discussion will focus on those areas most relevant to using the networks in the SNNAP environment. More details can be found in the references in each section and an overview of all three architectures is available in Wiggins, Looper, & Engquist (1991).

Back Propagation

Back propagation networks are the most widely and successfully applied network architecture[1]. They have been employed in numerous areas and their performance has been compared to many traditional clustering, pattern matching, and statistical techniques (for a review, see Wiggins, 1990). The success of back propagation in other areas of research and model building has recently been extended to personnel models (Wiggins et al., 1992). While the two other architectures supported by SNNAP have been successfully applied, back propagation networks have proven superior in all personnel research to date.

Back propagation networks utilize a layer of functions to develop relations between the inputs and outputs of a model. By using the output of some function as inputs into other functions, complex functional forms can be generated. Typically these functions are arranged in layers, with the first layer receiving its inputs from the inputs to the model and each succeeding layer receiving inputs from the prior layer. This continues until the output layer is reached, and this layer produces the output (or outputs) of the model. When all connections between functions proceed from input to output, the network is referred as a feed forward network. If connections are allowed back toward the inputs, the network is referred to as recurrent. In neural network terminology, the functions are referred to as neurons. A very simple example, using airmen reenlistment, is shown in Figure 1.

---

[1]Technically, back propagation is a term applicable only to the process of training networks, however, we will follow the convention of applying the term to the entire network architecture.

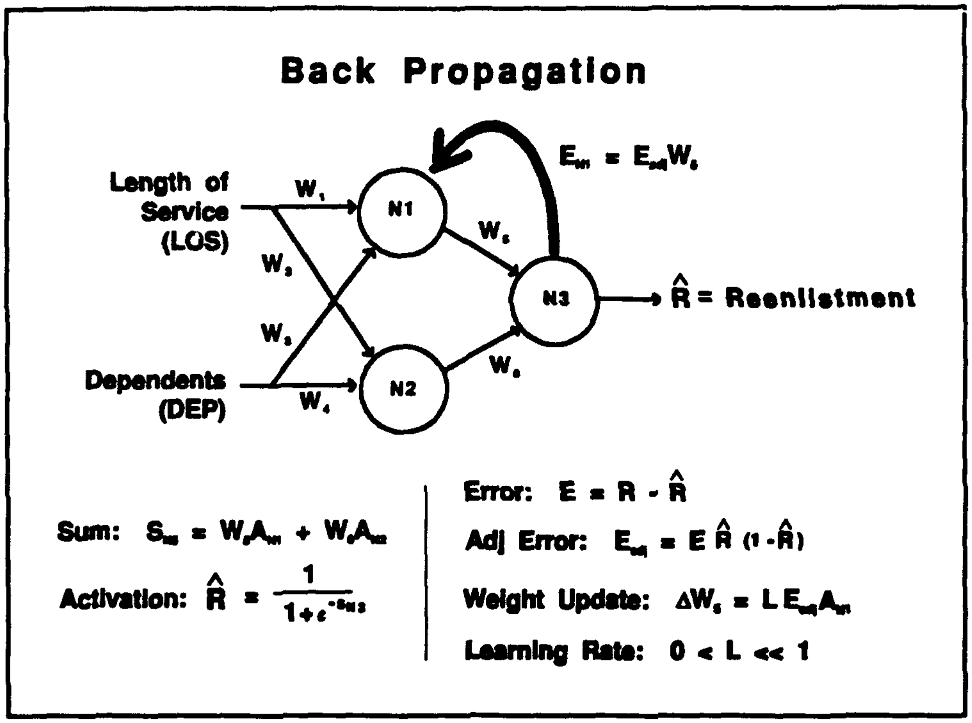


Figure 1.
The back propagation method (reenlistment example).

In Figure 1, two inputs (length of service and number of dependents) are used to model the probability an individual will choose to reenlist. The network has a very simple structure with two neurons (N_1 and N_2) in the first layer and a single output neuron (N_3) producing the modeled reenlistment probability. The arrows represent the flow of information through the network as the two neurons in the first layer "feed forward" into the output neuron. The first layer in this network is typically called a hidden layer because it does not have direct contact with any inputs or outputs. Back propagation networks usually have one or two hidden layers.

The weights or function coefficients are designated by the W_i terms in the figures. Back propagation neurons are usually modeled as simple inner products between the inputs and the neuron weights with the result passed through a nonlinear transformation (or activation function). The most common activation transformation is the sigmoid or logistic curve (which is computed in Figure 1). Hyperbolic tangents, a form of the sigmoid curve which is symmetric about 0 and ranges from -1 to 1 (the sigmoid ranges from 0 to 1) is also commonly used. Both of these activation functions are supported in SNNAP. Because these two functions have a limited range, they cannot serve to model all possible model outputs. SNNAP provides a linear activation function primarily to be used on output neurons in these cases. It has been proven that the structure of back propagation networks with either nonlinear activation function can support any smooth nonlinear mapping between the model's inputs and outputs (Hornik, Stinchcomebe, & White, 1989).

SNNAP includes a fourth activation function which is particularly well suited to capturing interactions among model inputs. For example, if the importance of length of service is changed

by changing the number of dependents. This type of neuron was recently suggested by Durbin & Rumelhart (1989) and does not use the standard inner product computations. The product unit has the following form:

$$O = \prod W_i I_i \tag{1}$$

Where:

O is the output of the neuron.

W_i are the neurons weights.

I_i are the inputs to the neuron

The form of the product unit makes its first derivative approach infinity as any input approaches 0. For this reason, it can only be used in the first hidden layer of a network and all inputs must be above 0 (methods of ensuring this condition are provided in SNNAP). Despite these restrictions, the product unit can improve the performance of networks in some problem domains using the training heuristics employed in SNNAP.

During a training process, the weights in the network are changed to improve the ability of the network in predicting the observed outputs from the supplied inputs. Usually the weights are adjusted in an attempt to minimize the sum of squared errors over the observations or exemplars in the training data (SNNAP utilizes this criterion). The actual weight adjustment is made adaptively by successively presenting each training exemplar to the network and adjusting the weights slightly to improve performance on that single exemplar. A clever application of the chain rule of derivatives (see Rumelhart, Hinton, & Williams, 1986) allows the errors at the output layer to be propagated back to the hidden layers. The entire process proceeds to minimize the sum of squared errors using gradient descent over the entire network weight space. This adaptive process is performed many times for each observation in the training set and a single pass through the training data is termed an epoch.

Two parameters determine how training proceeds in a back propagation network — the training rate and the momentum factor. The training rate essentially determines how much of the network's error is attempted to be solved by each weight being adjusted in the network (assuming a linear effect on error from the gradient). This value is usually set between 0.001 and 0.9 (although both lower and higher values can sometimes be used). Settings which are too high can cause in-sample network performance to degrade during training as weights compete to explain too much of the error. The momentum term works to smooth the network's training path by remembering past weight adjustments (see Rumelhart et al., 1986). The use of a momentum term can significantly increase training rates. With hundreds or even thousands of

training epochs required for back propagation, this improvement can mean hours of training time. In general, the training rate and momentum work together. A larger momentum term implies a smaller training rate should be used. This follows from the fact that the momentum term actually allows a weight adjustment to be carried forward over several observations. The momentum should never exceed 1 as this implies an exponential impact on training. The form of this impact is an infinite series and the total effect of training with momentum is expressed by:

$$\textit{Total Training Rate} = \frac{L}{1 - M} \tag{2}$$

Where:

L is the training rate

M is the momentum factor

SNNAP allows both recurrent and feed forward back propagation networks to be specified and trained. While feed forward networks are used for most applications, recurrent networks are particularly appropriate for time series data or other problems with a structure in time. The recurrent connections in the network allow the development of an internal structure relating current outputs to a representation incorporating both past and current inputs. The implementation of recurrent back propagation in SNNAP is a form of the simple recurrent network (SRN) developed by Elman (1990).

Probabilistic Neural Networks (PNNs)

A second major class of neural networks implemented in SNNAP is based on the estimation of probability density functions (PDFs) from the training data. These networks were first developed as a classification technique for problems where one must identify a binary or categorical outcome (e.g. reenlist vs. separate vs. extend). The networks have been extended in SNNAP to allow PNNs to work with continuous output variables.

Classification PNNs. As originally developed by Specht (1990), the PNN uses PDFs developed for each class or category into which exemplars are to be separated. The PDFs are generated using the kernel methods developed by Parzen (1962) for univariate distributions and extended by Cacoullos (1966) to multivariate distributions.

The PNN develops PDFs in the input space by placing a gaussian kernel (a pseudo-distribution which does not integrate to 1) over each observation in a set. These kernels are then summed to produce a PDF for the class. This process can produce distributions of virtually any shape. The smoothness of the distribution is determined by the assumed variance of the kernels placed over each observation. This variance is usually referred to as the smoothing factor for PNNs. The effect of different smoothing factors on a simple one-dimensional distribution can

be seen in Figure 2. Each of the distributions shown in the figure were derived using different scaling factors from the same 5 data points. Specht suggests that network performance is not dramatically affected by relatively large changes in the smoothing parameter. Computation of the PDFs is covered in detail in Specht (1990) and Wiggins et al. (1992).

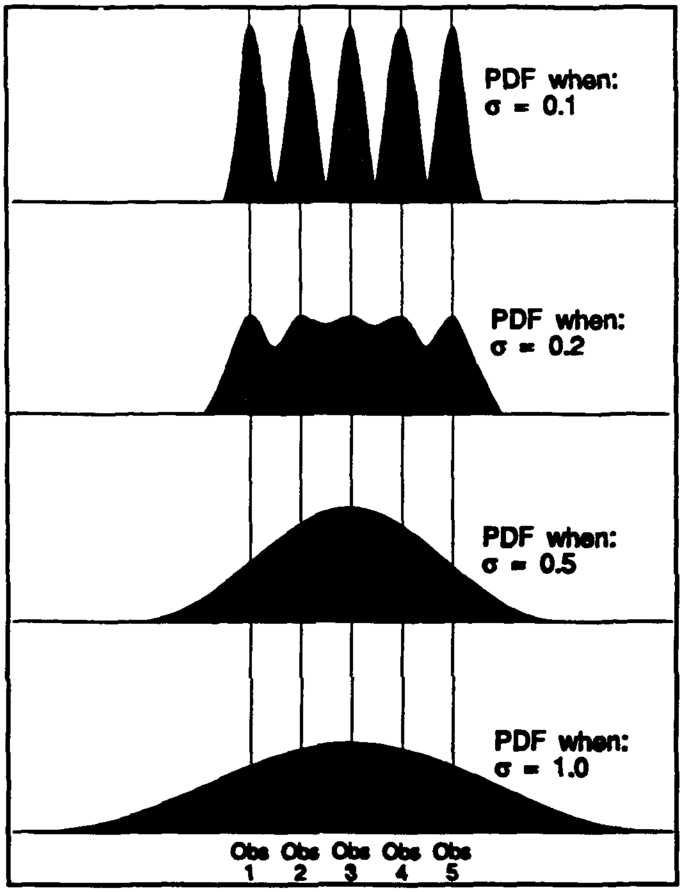


Figure 2.
Examples of PNN Gaussian kernels.

For classification networks, SNNAP implements a facility to choose the optimal smoothing parameter for a given training data set using hold-one-out methods (Weiss & Kulikowski, 1991). The class of each training exemplar is predicted using all of the other exemplars in the training data set. SNNAP uses a search procedure to find the smoothing factor which minimizes the sum of squared errors when predicting the estimation sample one exemplar at a time.

Once a PDF has been generated, a new exemplar can be selected into one of the classes based on the relative heights of the class PDFs when evaluated at the input values for the new exemplar. The class with the highest point density is selected as the most likely class for the new exemplar. This process can also involve a priori weights applied to each of the classes. SNNAP supports this weighting and uses the relative proportion of training exemplars in each

class as the default a priori weights. SNNAP also extends the classification process to produce the probability (based on the PDFs) of a new exemplar falling into each of the possible classes.

Continuous PNNs. SNNAP includes some extensions to the PNN architecture which were suggested by Specht (1990) and allow the network to work with continuous output variables. Conceptually, this process operates by developing a single PDF where the output variable forms one of the dimensions. To evaluate a new exemplar, the values are fixed for all of the known variables. This leaves a slice of the original distribution in the output dimension. This slice is a pseudo-distribution (which can be made a standard distribution by scaling) for the output variable given the input variable values. The most likely output value is then determined by finding the maximum likelihood point on this pseudo-distribution. This process can be extended to multiple outputs where the pseudo-distribution itself becomes multivariate; however, the computational burden becomes too great to be useful at that point.

PNNs for Probability Density Functions (PDFs). SNNAP implements a third variant of PNNs which is used primarily to support analysis of the other networks. This network uses the PDF directly to estimate the relative density of data in any area of input space. This allows the analyst or researcher to determine if the estimation sample contains sufficient data in an area of the response surface which is of interest. If little training data exists in an area of input space, this reduces the confidence in the projected outcome.

Learning Vector Quantization (LVQ)

The learning vector quantization (LVQ) network was developed by Kohonen (1984) and is also a classification network. The network has been applied to several problems and has often proven superior to standard classification techniques (Kohonen, Barna, & Chrisley, 1988). In several personnel areas, Wiggins et al. (1992) found the LVQ to improve on the performance of regression and probit models but to perform somewhat worse than back propagation models. In general, the LVQ requires considerably less training time than back propagation and this may be a factor in some cases.

The LVQ network bears a strong resemblance to the K-means clustering algorithm (Duda & Hart, 1973), but has some features which improve its performance in classification tasks. The LVQ network operates by generating a set of reference vectors (or neurons) and placing them in the input space. These reference vectors are located at points in the input space and serve as attractors for all exemplars which fall in their neighborhood. This can be seen in Figure 3, which shows a simple reenlistment model. In the top of the figure a hypothetical distribution of reenlisters and separators is shown. In the bottom of the figure, six reference vectors are placed in the two dimensional input space (3 to reenlistment and 3 to separation). Each reference vector has an area of influence within which all exemplars are assigned to the vector. A new exemplar to be projected is assigned to the nearest reference vector (usually computed by the Euclidian distance).

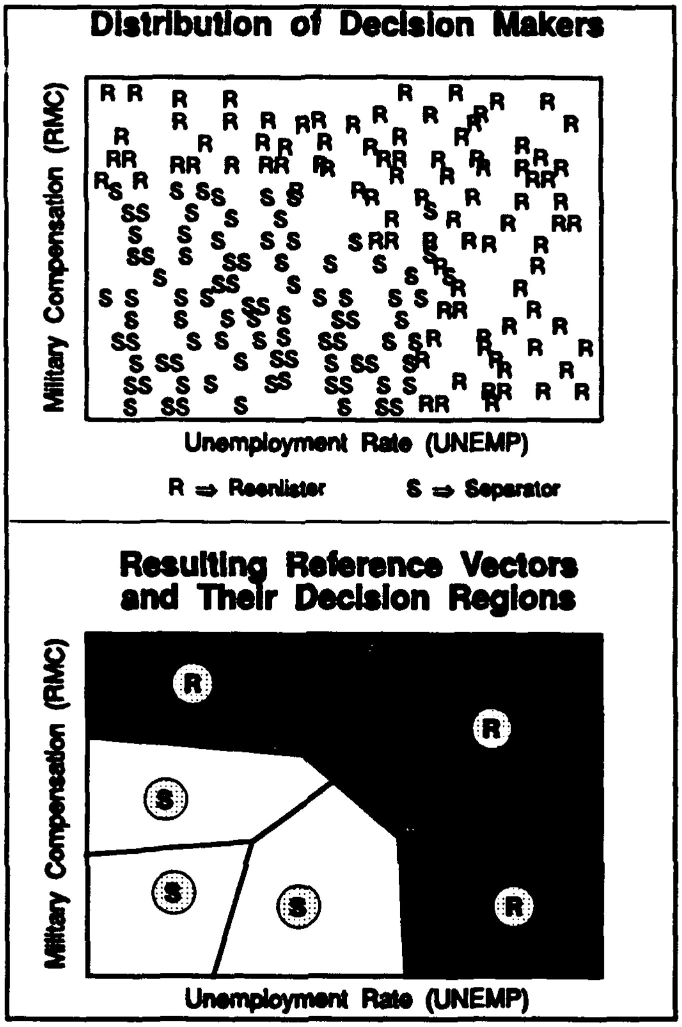


Figure 3.
Decision boundaries formed by an LVQ network.

Training in an LVQ network involves determining the locations of the reference vectors in input space. If these locations were chosen to minimize within exemplar input variance and maximize between exemplar input variance, LVQ would exactly reproduce the K-means results. However, LVQ uses the actual classes of the training data exemplars to determine optimal class separation boundaries[2].

The primary parameter which must be designated with the LVQ architecture is the number of neurons or reference vectors. In general, this number can fluctuate over a fairly wide range and produce reasonable results. SNNAP's expert system is also configured to suggest a number of neurons given the problem type and number of training exemplars.

[2]The details of these computations are provide in Kohonen (1984).

Data Handling

Specifying Data Elements

In order to analyze a data set, SNNAP must be able to read the data from disk and identify the fields containing input and output variables. Three different types of data can be used by SNNAP: fixed format, free format, and delimited data. In fixed format files each input and output variable is found in a specific column and each physical disk record represents an observation. Free format files use spaces or tabs to separate each variable. In this case, each record need not represent an observation. The variable list is looped through as each field is encountered, such that an observation may span several lines in the disk file. Delimited data is similar to free form data except the user can specify a specific character (such as a comma) which separates each field. Two delimiters in a row will be interpreted as a zero in the appropriate field. Again, with delimited data, an observation may span several lines. For all three types of data a format file is used to specify variable names and identify the type of variable (numeric, categorical, binary). See Appendix A for a complete description of the format file.

Scaling and Standardizing

The training algorithms for most neural network architectures are highly susceptible to the scale of the input variables. In particular, they can be affected by differences in scale among the input variables. To address this problem, SNNAP has the capability to automatically scale the input variable to lie within any range specified by the user. Normally such ranges would be small (between 10 and -10). In fact, it is common to scale the range from 0 to 1 for sigmoid networks and -1 to 1 for hyperbolic tangent networks. The product units which are allowed on back propagation networks require their inputs to be positive. In this case the minimum value for the scale should typically be at least 0.1. An alternative to a user specified scale (in all cases except product units) is to standardize the data to mean 0 and standard deviation 1. SNNAP will automatically perform this standardization for all inputs and this is usually the suggested default.

It can also be useful to scale the output variable or variables. The back propagation algorithm often trains faster if all of the network's neurons use a transfer function with a similar range. However, for example, a sigmoid output neuron will only produce values in the 0 to 1 range. If the actual output ranges from -100 to 10,000, a 0 to 1 output will always be a poor approximation. SNNAP will allow the output value to be automatically scaled for internal use by the network so that a sigmoid (or hyperbolic tangent) function can be used as the network's output neuron. Alternately, a linear output neuron (which has an infinite range) could be specified. However, this often slows network training significantly.

Hold-Out Sampling

The ability of neural networks to produce complex and nonlinear relations between model inputs and outputs is one of their greatest assets. However, this ability can cause problems if the training data set contains a large stochastic component (i.e. the data has a large unexplained component or is noisy). When confronted with a noisy training data set, a neural network has the capability to "memorize" the noise in the data. Noisy training data leads to a problem similar to over-fitting with regression models containing high order terms. The network's performance is very good in-sample (often flawless); however, when confronted with data not in the training data, the network performs very poorly.

This ability to perform out-of-sample is referred to as the generalization problem. In all studies performed on personnel data, some method of preventing over-fit has been absolutely essential in developing models which generalize outside of the training sample (Wiggins et al., 1992). The problem can be easily visualized using an example with back propagation training (see Figure 4). Back propagation is an adaptive process and requires many passes through a data set (epochs) for the network model to complete training. With slow training rates, performance always improves within the training sample. However, if performance is tracked on a hold-out or validation sample, this performance may degrade significantly beyond a certain point in training.

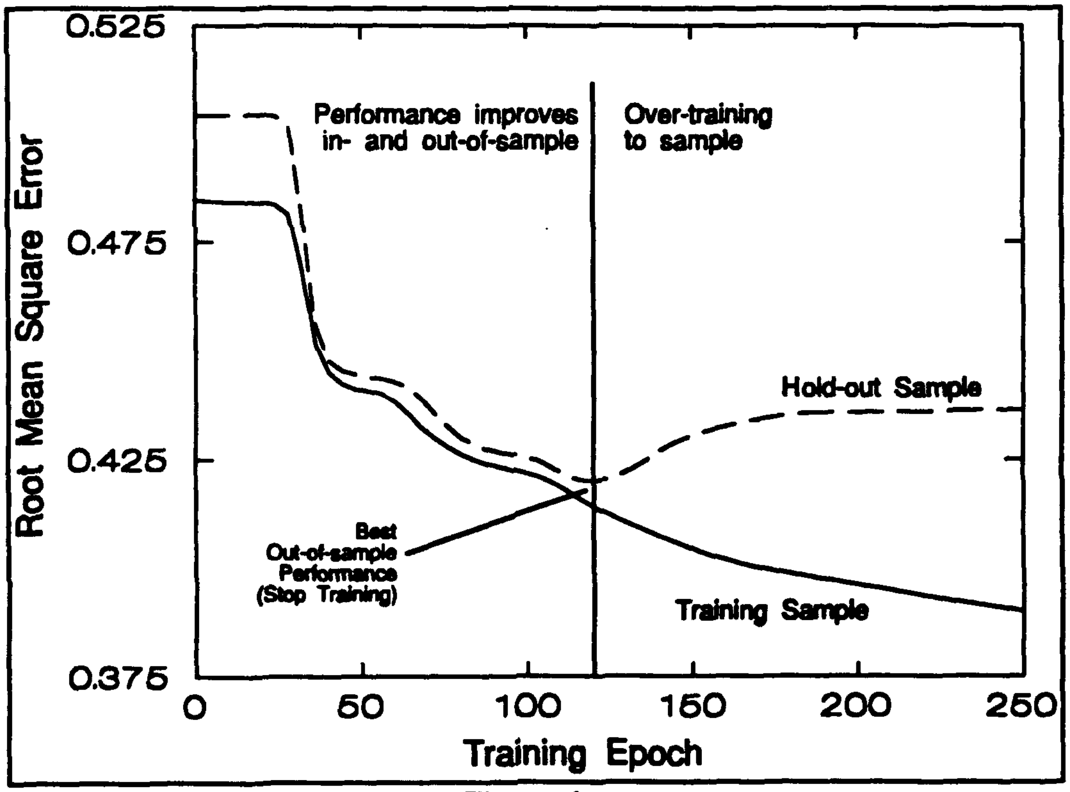


Figure 4.
Training path for back propagation.

SNNAP provides facilities for saving a copy of a back propagation network each time a hold-out sample error basin (such as the one in Figure 4) is encountered during training. This is an extension to the early stopping training heuristics suggested by several researchers (Wiggins et al.; 1991; Morgan & Bourlard, 1990; Rumelhart, 1990). In the simple example shown in Figure 4, the hold-out sample performance (dashed line) has a single minimum point. In practice, several minimum "basins" can be encountered and the researcher would usually choose the one with the smallest root mean square error.

In addition to improving the predictive capability of networks, the performance on a validation sample provides some measure of confidence when interpreting the relations the network displays between model inputs and outputs. Standard statistics employed with regression models are not always applicable to neural networks and the extremely flexible form of network architectures makes in-sample performance statistics meaningless. Hold-out or validation sample performance provides a quantitative measure of a network model's predictive ability.

Variable Summary

SNNAP provides a facility to obtain simple statistics on the variables in an analysis. Statistics include the mean, standard deviation, minimum, and maximum values for any variable. These values can be useful in determining the appropriate range over which a model's response should be evaluated. When projecting with a model, they also indicate whether the model is operating within the bounds of the training data or is extrapolating in a region where no estimation data was available. As discussed later, the facility which provides these statistics also plays a direct role in network analysis.

Automatic Parameter Selection

An expert system is embedded within SNNAP to assist in selecting the structure and parameters for each network type. This feature appears as a **Suggest** button on the dialog box where a network's structure is determined. The "suggestion" made by the expert system is based on the number and type of variables being analyzed, the size of the data set, and the results of prior research using neural networks on personnel data. Any aspect of a network determined by the **Suggest** option can be modified by the user. This facility simply provides a base network to which changes can be made.

Different structure factors and parameters are used for each network type. The primary structure considerations determined for a back propagation network are the number of neurons in the hidden layer, the type of neuron transfer function, and whether recurrent connections are employed. When the training performance is tracked on a hold-out sample, the structure of the network has been found to have little effect on out-of-sample network performance. If the network contains a sufficient number of neurons to represent the relations in the data, additional

neurons have little detrimental effect[3]. For this reason SNNAP typically suggests networks with more neurons than may be required. Default values for the training rate and momentum factors are also specified.

For LVQ networks, the primary structural factor determined by **Suggest** is the number of neurons (or reference vectors) used by the network. Default training rates, conscience factors, and number of epochs to train for each training phase are also provided. PNNs require even less structural information. The authors suggest that the data for PNNs always be standardized. Given this standardization, a single default smoothing parameter is appropriate and for PNNs the **Suggest** option does very little.

Network Training

Complete facilities are provided for training the back propagation, PNN, and LVQ network architectures discussed earlier. In addition, extensive error reporting is available for the back propagation and LVQ architectures. Both of these architectures are adaptive and make many passes through a training data set before converging. Both the training and validation sample performance can be tracked while this training occurs. If the network has multiple outputs, any or all of the outputs can be tracked.

As discussed earlier, SNNAP implements training heuristics to make copies of the network whenever hold-out sample performance reaches the bottom of an error basin during training. The copies contain the complete state of the network at the specific point during training: network weights, momentum factors, current outputs, etc. These copies of the network are named and can be selected later for further training or for analysis. In fact, one of these saved networks is typically selected as the final model. In addition to the hold-out error basin heuristic, SNNAP allows copies of the network to be made whenever training goes from being increasingly easy to more difficult (the second derivative of in-sample error goes from positive to negative). This inflection heuristic has also proven useful in selecting models which perform well out-of-sample and does not explicitly require the use of a hold-out sample (Wiggins et al., 1991; and Rumelhart, 1990). However, there can be multiple occurrences of this inflection point and an understanding of the errors on a hold-out sample can assist in choosing the appropriate inflection point.

Training can be stopped at any point and training parameters changed. Training can then be re-started from the point where it was stopped. All network analysis facilities discussed below can be performed while a network is being trained. The current status of the network model will be reflected in the selected analysis or view.

---

[3]This is clearly not the case when hold-out testing is not done. In this case, network structure serves as the primary means of producing valid generalizations. SNNAP does not attempt to make appropriate network structure suggestions for training without hold-out samples.

Network Analysis and Views

Analyzing and visualizing the response of neural network models is one of the strongest elements of SNNAP. These features are critical if one hopes to explicate the nonlinear features in a successful neural network model[4]. With the possibility for interactions among the input variables and nonlinear impacts on the output variables, neural network models are not easy to summarize. Simple, fixed effects coefficients or elasticities will rarely represent the structure of a network model. Instead, the effect of an input variable on the output variable may depend on the level of the input variable and even the level of other input variables. This gives the network model a potentially rich structure on which to base projection. However, it provides an equally rich environment for model analysis if the appropriate tools are applied. Toward this end, SNNAP provides several facilities for displaying and analyzing network response surfaces.

Graphical Views

The cornerstone of the network analysis facilities in SNNAP is the ability to generate views of the network's response surface. One of the most intuitive ways to perceive a network's behavior is with graphical views of the response surface. SNNAP provides facilities to easily generate 2 and 3-dimensional graphs of an output's response to various levels of one or two inputs (as modeled by the network). As will be shown in the example section, extensive control is provided over the appearance of the graphs. Facilities are also provided to view the graphs with logorithmic transformations of any of the variables or to view the derivative of the output with respect to any of the inputs. These derivative or marginal effect graphs can be particularly useful in showing any change in the impact of inputs as any input changes level.

A network with only 2 inputs could be completely described by a 3-dimensional graph. For models with more than 2 inputs, the graphs represent slices of the response surface where all other variables are held at constant values. SNNAP allows these other values to be set such that various slices of the response surface can be presented. In all cases, multiple views of one or several networks can be presented on the screen simultaneously. In fact, views can be made on a network while it is training to evaluate the current training point.

Tabular Views

The graphical views of the response surface can be toggled at any time to a tabular view of the same information. By adjusting the range and frequency at which samples are taken, these tabular views can cover any response area of interest. The tables provide a reference for the graphical views.

---

[4]By their very nature, successful neural network models are nonlinear. If the underlying phenomenon is linear or has a known nonlinear form, the best possible model can be developed by specifying the known form and selecting model coefficients based on some criterion. In these cases, neural network models cannot exceed the performance of traditional techniques. They can only serve to reinforce the modeler's assumptions about the model's structure.

Model Performance, Statistics

Several model performance statistics can be generated for any neural network model. These statistics include the Root Mean Square Error (RMS); Theil's inequality coefficient (TIC) and its bias, variance, and correlation components; the simulation R-squared, the Janus Quotient, and the correlation between actual and predicted outputs. Each of these measures was summarized and documented in a prior publication (Stone, Looper, & McGarrity, 1990). In addition, SNNAP computes the means and standard deviations of the actual and predicted outputs. All statistics are available both for the training sample and the selected validation sample or samples. These statistics provide a means of comparing the in- and out-of-sample performance of different models and evaluating the performance of a single model.

Comparative Models, OLS

SNNAP provides an option for estimating ordinary least square (OLS) regression models. These models are treated in the same manner as the neural network models with complete access to the sub-sampling, performance statistics, and views. In many cases, the OLS models provide a good baseline for evaluating the performance of a neural network model. Even if OLS is not the appropriate technique for a specific problem, such as a binary decision problem, it provides some test of the network models' relative performance.

Automated Response Surface Scanning

The response surface of a neural network can be difficult to analyze even with the tools just mentioned. To provide for an initial analysis of the response surface, SNNAP can automatically search the response surface of a model. This is done by visiting the surface at each observation in the training sample. The first and second derivative response of the network is noted at each point and nonlinear or interacting features are detected. Variables which have little impact on the output variable are noted. Specific functional relationships between the inputs and the output are searched for: linear, log-linear, linear-log, and log-log. Cases where the impact of one input depends on the level of the input or other inputs (interactions) are also sought.

The sensitivity of this search can be set by the user. This sensitivity determines, for example, what range of response will be interpreted as linear. The user can also determine the range over which the search is performed. By default all training observations are visited; however, it is sometimes preferable to search on areas where training data is most dense.

Saving and Restoring

Save and restore capabilities are available for all objects in the SNNAP environment. Networks can be saved at any point during training and later restored to their exact condition. Additional training or analysis can be performed at that point. Graphs and surface scan (searches) results can also be saved and restored.

DETAILED EXAMPLE: AIRMAN PERFORMANCE

The Airman Performance Problem

An analysis of airman performance and its relation to aptitude and experience will be used to demonstrate SNNAP facilities and their application to a specific problem. The approach will focus on using SNNAP to analyze the problem rather than theoretical, institutional, or data considerations. Most of the facilities available in SNNAP will be demonstrated and several other options will be discussed.

Following the work of Lance, Hedge, & Alley (1987) and Vance, MacCallum, Coovert, & Hedge (1989) this example will be based on walk through performance test (WTPT) results. The WTPT is an objective measure of performance based on the ability to correctly complete critical steps in performing a specific task. At the Air Force Specialty (AFS) level, WTPT evaluated eight specialties across several tasks with trained observers evaluating the performance of each step within each task.

This example will focus on a single task in AFS 324X0 (Precision Measuring Equipment Specialists); more details on the WTPT methodology can be found in Hedge (1984) and Hedge & Teachout (1986). Specifically, hands-on performance on the task "Calibrates Distortion Analyzers" (designated H645) is analyzed in this example. The proportion of task steps performed correctly is used as the performance metric (task H645 is 30 steps which are listed in Appendix C).

As a measure of aptitude, all four of the Selector Aptitude Index (AI) scores are used. These four scores are composites of the 10 Armed Services Aptitude Battery (ASVAB) sub-test scores. The number of times an airman had performed the "Calibrates Distortion Analyzers" task is used as a measure of task specific experience. This experience value was self-reported by the job incumbents when the WTPT was administered. All of the variables used in the analysis are summarized in Table 1. Complete information on these variables was available for 124 of the 140 airman administered the WTPT. The basis for model development will be these 124 cases with 1 output variable and 5 input variables.

The process of creating models and analyzing the data with SNNAP is addressed below. The model runs under the Microsoft Windows 3.0 or 3.1 environment and the user is expected to be familiar with the operation of the Graphical User Interface. Some aspects of the interface are briefly explained, but a knowledge of standard menus, dialog boxes, and drop-down menus is assumed. New windows users should refer to the Windows Users Manual or the on-line help for more detailed explanations.

Table 1.
Variables in the Performance Model

Variables	Descriptions
H645per	Percent of steps completed correctly on the "Calibrates Distortion Analyzers" task (output/dependent variable).
Mp	Mechanical selector AI percentile
Ap	Administrative selector AI percentile
Gp	General selector AI percentile
Ep	Electrical selector AI percentile
H645num	Number of times the "Calibrates Distortion Analyzers" task was performed by the job incumbent prior to the WTPT.

Getting Data into SNNAP

Before proceeding with an analysis of the performance on the task, the data format of the data and variable names must be provided to SNNAP. As discussed earlier, three types of files can be read by SNNAP: fixed format, free format, and delimited. The process for specifying the files is basically the same and the use of fixed format files is described here.

Using Fixed Form Data

SNNAP requires that all data files have a format file which describes the contents of the data file. By convention, this file always ends with the .FMT suffix and can be prepared in any editor or word processor which can produce ASCII files. All three types of files utilize the same type of format file although some fields are ignored for free format and delimited data files. See Appendix A for a complete description of a format file.

Specifying the Format File

When starting SNNAP from a clean slate, the first operation is to create a new network. This process is initiated by selecting the New option under the **File** menu in the main menu bar as shown in Figure 5. Alternately, the right mouse button may be clicked to produce a pop-up menu which contains the New and **Save** options. In most cases, SNNAP options may be invoked from the main menu bar or from a context sensitive pop-up menu which contains the most frequently used commands for the currently active window. This pop-up menu is always obtained by pressing the right mouse button.

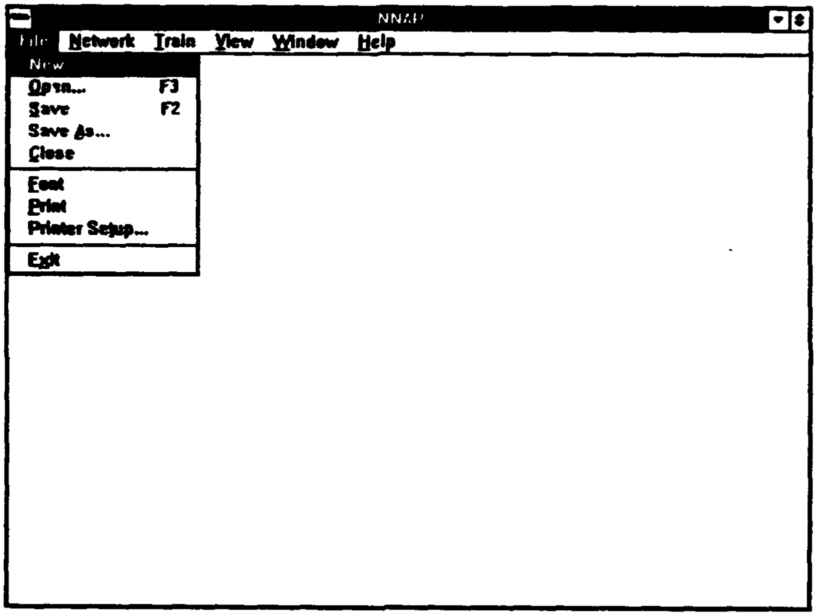


Figure 5.
Starting the process to create a new network.

The first dialog box under the New option allows the user to select the format file for the data set to be analyzed. In this case, the format file s645.FMT is selected with the mouse by clicking on the name in the Files menu box. The OK button is then selected. (Alternately the s645.fmt name may be double-clicked.) Figure 6 shows the format dialog box after the s645.fmt file has been selected.

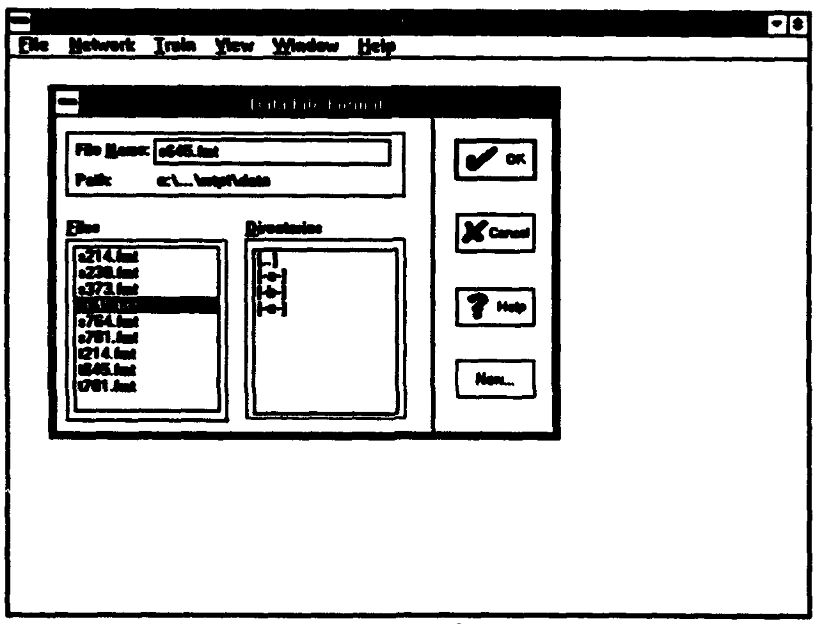


Figure 6.
Specifying the data format file.

Specifying the Variables

Following selection of the format file, the user is allowed to choose the variables from the data set which are to serve as the inputs (independent variables) and the outputs (dependent variables) for the model. As shown in Figure 7, the selected input variables are those discussed earlier and documented in Table 1. The output variable is the proportion of steps correctly completed in the hands-on portion of the "Calibrates Distortion Analyzers" task. Clicking on the **OK** button confirms the selected variables.

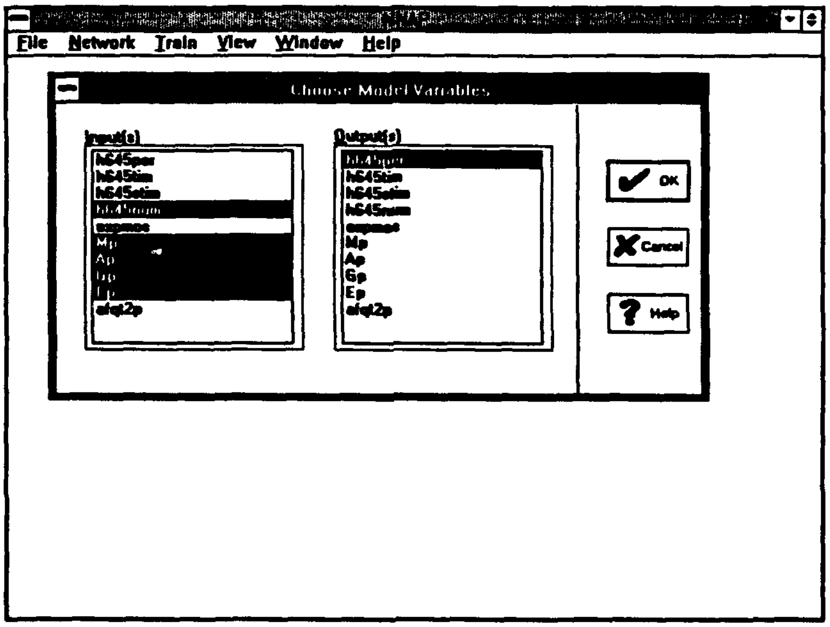


Figure 7.
Selecting the input and output variables for a network.

Selecting a Data Set and Sub-Samples

The next dialog box allows the user to select the data set on which the network model is to be trained. As can be seen in the **Files** box of the **Training Data** dialog box in Figure 8, the s645.dat file has been selected to train the network.

In the lower portion of the **Training Data** dialog box, the **Modulus** option has been selected for generating a hold-out or validation sample. The **Define Validation Sample** dialog box shows that a divisor of 5 and remainder of 3 has been selected for the modulus option. The 5 implies that every fifth case in the sample will serve as part of the validation sample. The three designates which of the five cases in each block of 5 is to be "held-out" (the 3rd observation). By selecting remainders of 0 through 4, any of 5 different hold-out samples could be generated using one fifth of the data as a validation sample. If an even split is desired between training and validation samples, the **Divisor** would be set to 2. In the current example,

the divisor of 5 implies that 100 (or 99) cases will be available for training and 24 (or 25) cases will be kept in the validation sample.

The observations designated by the modulus rule will not be used during training, but the performance of the network will be tracked on this sample to test performance. In addition to the pseudo-random selection with the modulus rule, SNNAP allows a separate file to be designated as a validation sample. This option is particularly suited to validation over different time frames and is available under the **Validation Sample 1** and **Validation Sample 2** drop down menus. SNNAP allows two different validation samples to be generated using either of the selection methods. None of the data in either validation sample will be used during network training.

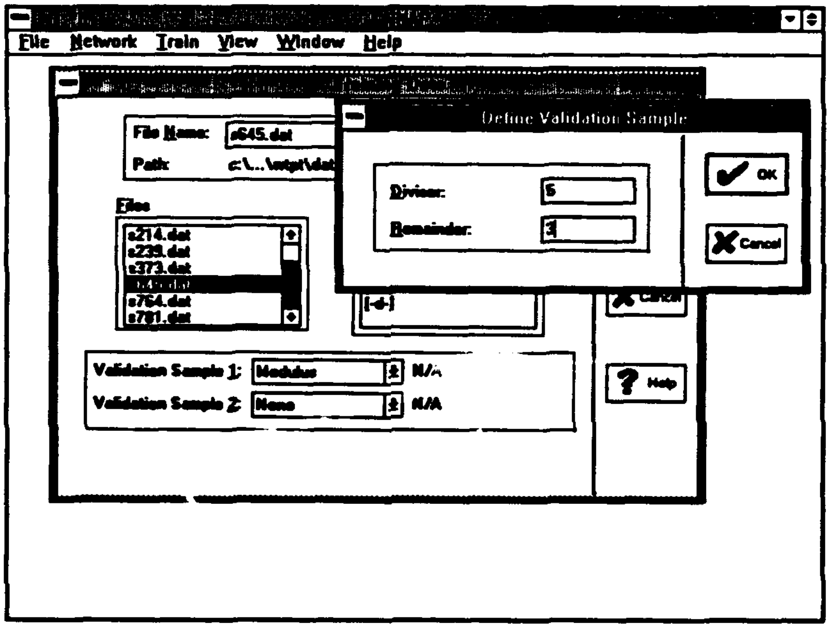


Figure 8.
Using modulus sub-sampling to designate a validation sample.

Generating a Base for Comparison: Least Squares

Before proceeding to the development of a neural network model of task performance, an Ordinary Least Squares (OLS) model will be estimated to provide a baseline for the network model. Some form of benchmark model is extremely important in applying neural networks as they provide no intrinsic statistics on their own performance. Knowledge of the in- and out-of-sample performance of a baseline model can also help in assessing the progress of neural network training.

Selecting the "Network"

In SNNAP, an OLS model is treated like a network model. In this way, all of the SNNAP's tools which have been developed for networks are directly applicable to the OLS models. The dialog box shown in Figure 9 for developing a **New Network** is the final sequence initiated when New was selected (for OLS models). As can be seen, the **Ordinary Least Squares** option is being chosen from the **Network Type** menu box. A title for the OLS "network" has been entered in the **Title** section.

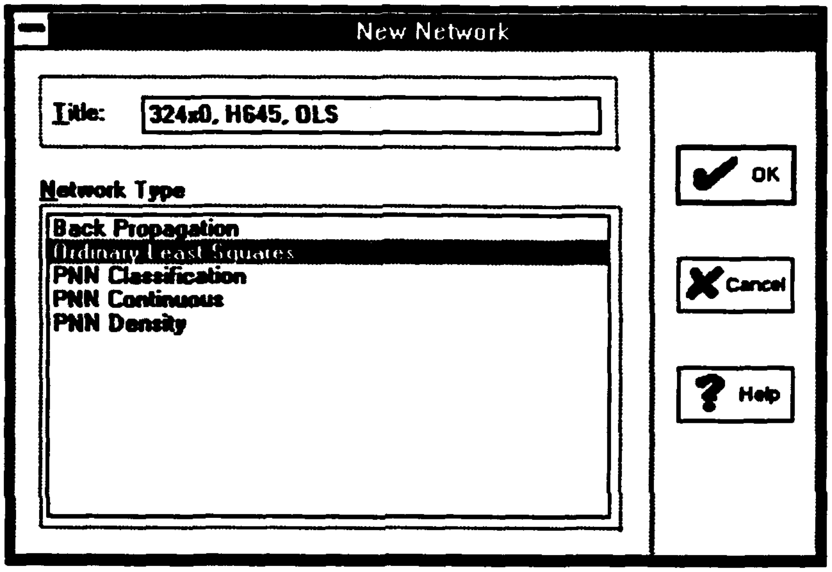


Figure 9.
Selecting the Ordinary Least Squares "network" type.

As shown in Figure 10, completion of the New process produces a network window (which in this case contains an OLS model). The window contains a header section which describes the type of model, the name of the data set, the types of validation samples, and the current epoch (used only for back propagation and LVQ training). A separate step (described below) is required to obtain the OLS estimates.

Getting a Data Summary

It is typically a good idea to briefly examine the data read by SNNAP using the format file to ensure that the correct variables are being read. When a data file is designated, SNNAP immediately reads the file to gather basic statistics used by several SNNAP facilities. These statistics are available to be viewed by the user. The option for viewing the statistics is the **Defaults** item under the **Networks** menu. However, it is shown being accessed in Figure 10 with a pop-up menu brought up with a right mouse click.

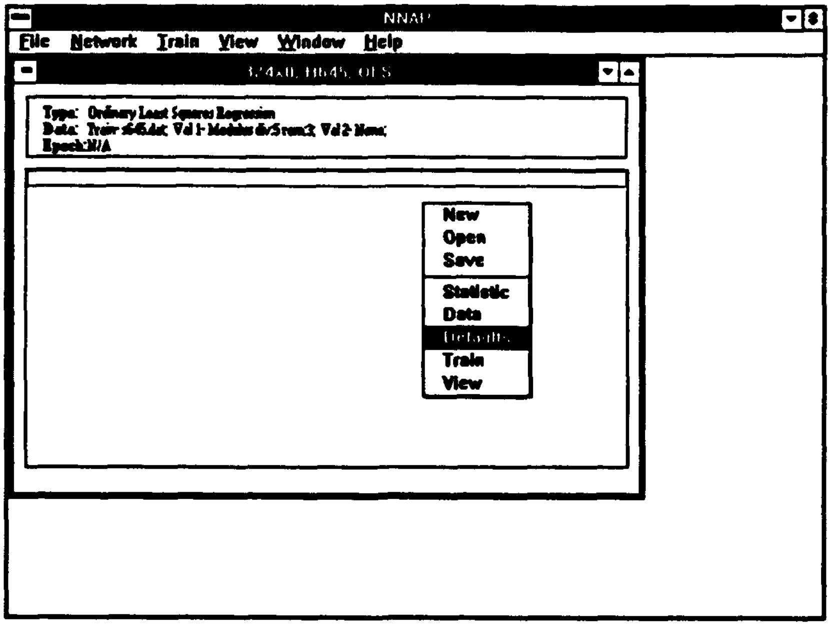


Figure 10.
Using the pop-up menu to examine summary statistics.

Figure 11 displays the operation of the **Defaults** options. When a variable is selected from the **Variables** menu box, its summary statistics are presented in the **Statistics** portion of the dialog box. Here the statistics for the Electronic percentile (Ep) are shown. This option actually provides a much broader service by allowing the variables in the **Values** area of the window to be modified. This use will be addressed later.

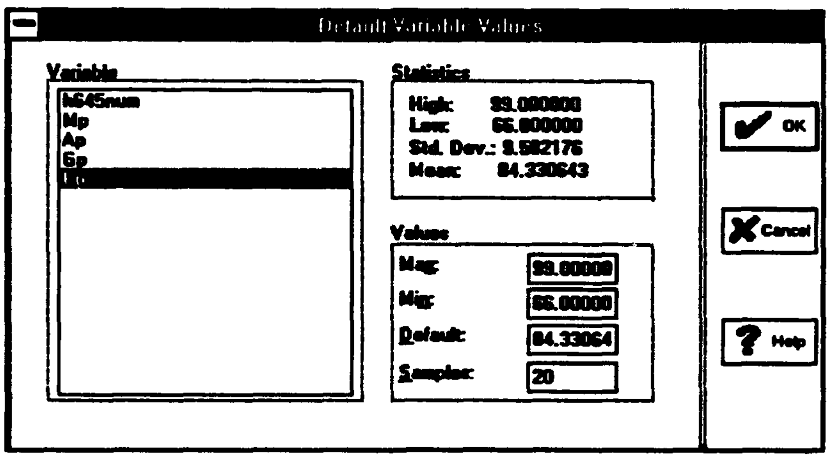


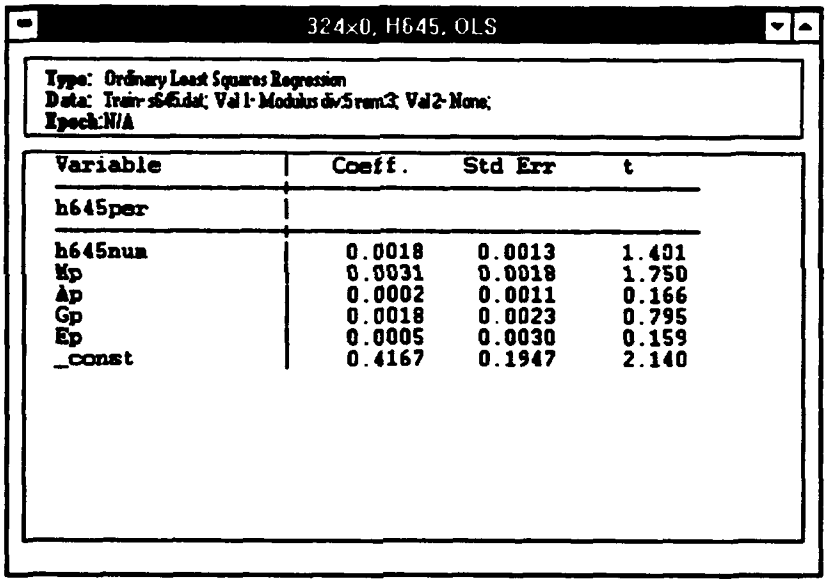
Figure 11.
The summary statistics for the Ep variable.

Least Squares Results

In order to perform the OLS regression, the **Train** option is selected from the **Train** menu on the main menu bar (or from the current pop-up menu). This step is required because

the OLS regression is treated like any network model. When **Train** is selected, the OLS regression results appear in the network model window (if the results exceed the size of the window, scroll bars can be used to scroll the window). In all cases, the OLS facility excludes the validation sample(s) from the estimation process. This behavior will be exploited later to compare OLS and neural network model performance.

As can be seen in Figure 12, the coefficients and their standard errors and t-statistics are provided by SNNAP. The OLS facility is not designed to be a full featured regression package, but to provide simple baseline comparison models for the neural network models. In should be noted that the OLS model need not use the same variables as the neural network models. In particular, many existing regression models apply logs, squares, or other transformations to their input terms (or output). While it is uncommon to apply such transformations to neural network inputs[5], separate variables containing the transformed data can be included only in the OLS models. This makes it possible to compare neural network performance against many existing models completely within the SNNAP environment.


324x0, H645, OLS

Type: Ordinary Least Squares Regression
Data: Train-s645.dat; Val 1-Modulus div5 rem3; Val2-None;
Epoch:N/A

Variable	Coeff.	Std Err	t
h645per			
h645num	0.0018	0.0013	1.401
Mp	0.0031	0.0018	1.750
Ap	0.0002	0.0011	0.166
Gp	0.0018	0.0023	0.795
Ep	0.0005	0.0030	0.159
_const	0.4167	0.1947	2.140

Figure 12.
OLS results for the airman performance model.

Developing a Back Propagation Model

With a baseline model in hand, we can proceed in developing a neural network model of task performance. For this example, the back propagation architecture will be used. This architecture has consistently shown the best performance in personnel research (Wiggins et al., 1992).

[5]Occasionally input transformations can be fruitfully applied with neural networks. If the model is of a circle or disk, a sum of two squares would make the problem much more tractable.

To a point, the back propagation model is specified in precisely the same manner as the OLS model just developed. The **New** option is invoked and the format, variable, data set, and sample selection steps detailed in Figures 5, 6, and 7 are performed. In this case, the exact same variables, data, and sum-samples were selected for the back propagation model. When the **New Network** dialog box is reached, the **Back Propagation** option is chosen from the **Types** menu box (see Figure 13). When **OK** is selected, the **New** process will proceed to the next step in specifying a network.

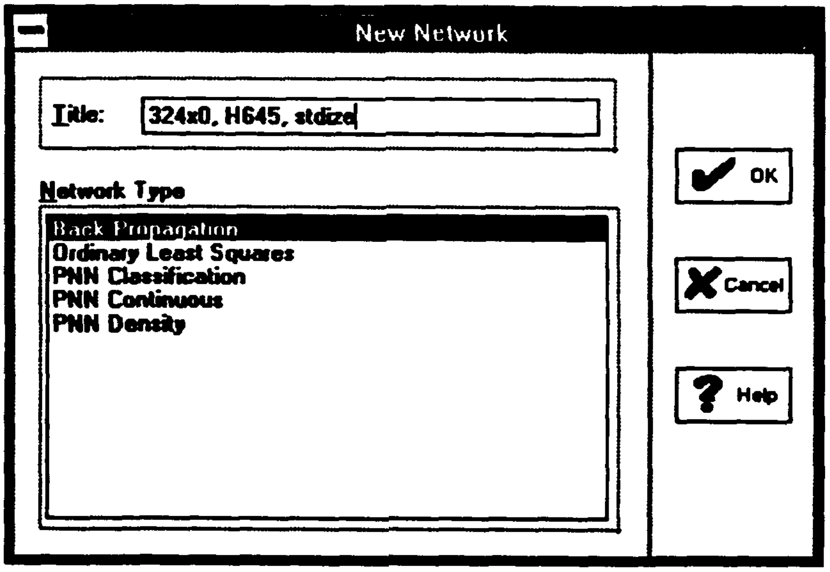


Figure 13.
Selecting the back propagation network type.

Using the Suggest Option

At this point, the **Structure** dialog box appears and allows the user to specify the structure of the back propagation network (see Figure 14). As discussed earlier, a back propagation network is usually composed of several layers which feed information forward from the input to the output layer. The **Structure** dialog box allows the user to set the number of layers, the types of activation functions, and the interconnections among layers.

The **Structure** dialog box also contains access to an expert system which will suggest a network architecture given the type of data specified and the size of the model. This facility is accessed through the **Suggest** button in the lower right corner of the dialog box. Suggest builds a "suggested" network structure which can then be examined or modified by the user. For the current model, the results of the **Suggest** were taken directly. A model with a single hidden layer of 15 neurons with sigmoid activation functions, and a sigmoid output neuron.

The model has no recurrent connections; e.g. layers which connect to themselves or layers closer to the input. Because there is no relationship between current training observations and prior observations in the training data set, such recurrent connections would be

inappropriate. If the data were generated by a time series, such relationships would likely hold, and recurrent connections could prove very fruitful[6].

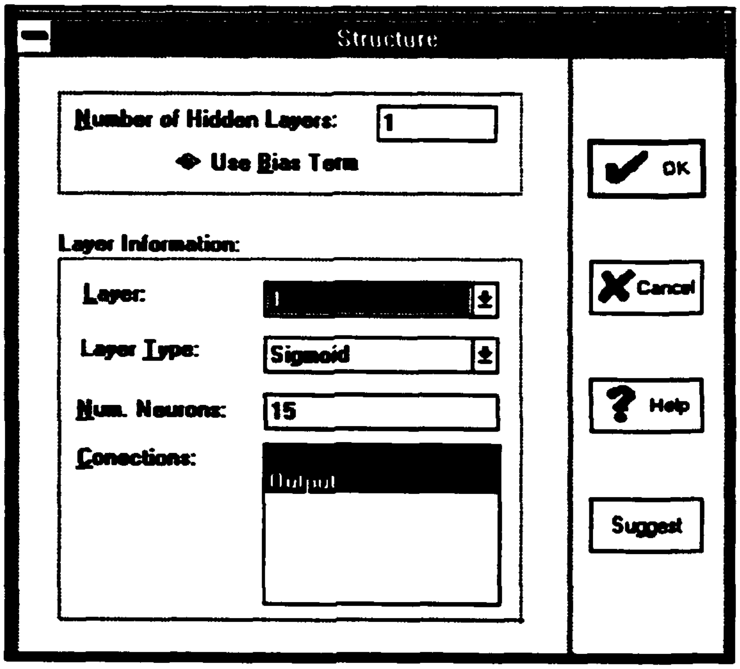


Figure 14.
Specifying the structure of a back propagation network.

It would also have been possible to build a network structure "from scratch". The number of hidden layers is specified at the top of the dialog box. When this number is designated or changed, the number of layers available in the **Layer** drop down menu changes appropriately (the input and output layers are always available in the menu). When a layer is selected from the **Layer** drop down menu, its activation function (**Layer Type**), number of neurons (**Num. Neurons**), and connection strategy (**Connections**) become available for editing. The **Layer Types** available include the linear, sigmoid, hyperbolic tangent, and product unit neurons discussed earlier. The highlighted connections designate the other layers into which the selected layer feeds its outputs. Virtually any connection strategy is possible.

Setting Parameters

Following the **Structure** dialog, the **Parameters** dialog box appears and allows the user to change the default parameters for network training. As seen in Figure 14, these parameters include the training rate and momentum factors discussed earlier. The range of the initial weights in the network can also be set. For the current example, all parameters are kept at their default settings.

---

[6]The expert system takes note of these possibilities when informed by the user and would "suggest" recurrent connections in such cases.

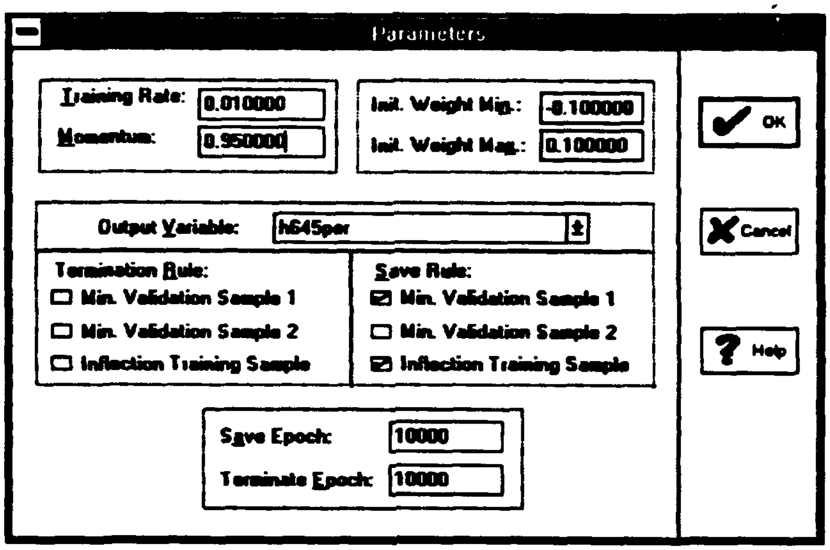


Figure 15.
Specifying training parameters and network save points.

Selecting Stopping and Network Save Points

Using the **Termination Rule** and **Save Rule** check boxes, this dialog box also allows the user to select when network training should be stopped and what rules are applied to save networks. For the current example, no termination rule will be selected. Instead, training will be manually stopped when it is apparent that further improvement in the validation sample is unlikely. In general, we would not recommend using rules to stop network training. It is not uncommon for the training path to contain several RMS basins for the validation sample.

Using the **Save Rule** check boxes, two different rules are applied to save copies of the network. Each change in the inflection of the training path generates a copy of the network at that point. In addition, each minimum or basin in the validation sample performance will generate a copy of the network. We will see later how these copies can be retrieved.

Using Data Scaling

As mentioned in the overview section, most networks train better if all of the input variables share a similar scale. Back propagation networks exhibit this characteristic. The **Data Scaling** dialog box which appears next in the New process allows the input and/or output variables to be scaled or standardized. As can be seen, we have selected standardized scaling for the input variables and no scaling for the output variables. To standardize each input variable, the mean of the variable will be subtracted and the result divided by the standard deviation before network training. Standardizing puts all of the input variables onto a relatively common scale. This operation is transparent for all view options where the variables are always re-transformed into their original range. In our example, there is no need to scale the output variable as a proportion naturally falls within the range of the sigmoid output neuron (0 to 1).

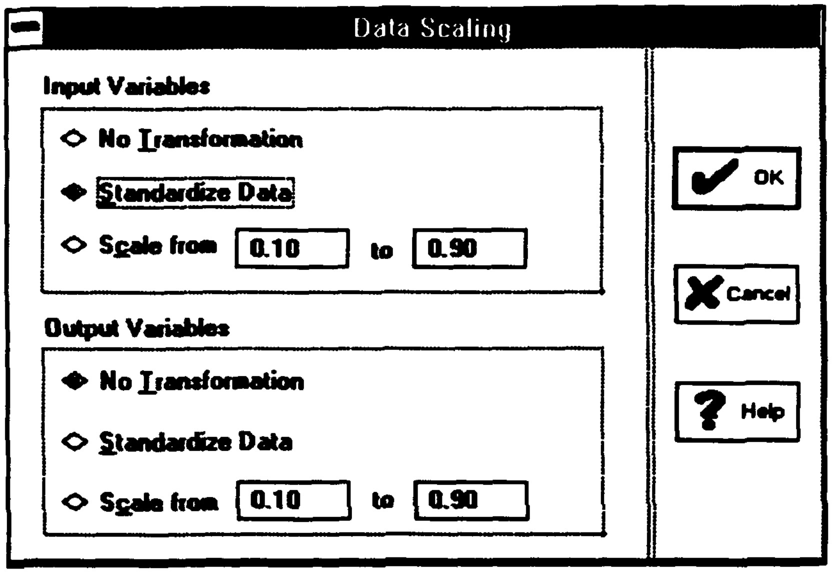


Figure 16.
Scaling or standardizing the networks inputs and outputs.

Changing Aspects of a Network

When this operation is complete, the back propagation network has been built and a blank network window appears (similar to the OLS window seen in Figure 10). At this point, or after some training, several aspects of the network can still be changed. The **Training Data, Parameters,** and **Data Scaling** dialog boxes can be accessed from the **Network** menu on the main menu bar or through the current pop-up menu when the network window is active. Once accessed, changes may be made to the network on any of these dialog boxes. Only the structure of the network is fixed. Using the **Structure** option, the structure of the network may be reviewed on a dialog box identical to Figure 14, however no changes may be made on this box. To change the structure of a network, the **New** option must be invoked.

Training

When the **New** process is complete, the back propagation network has been built and an empty network window similar to Figure 10 appears (only the summary information box contains different information). To begin the training process, the **Train** option is selected from the **Train** menu or from the current pop-up menu. Back propagation training will proceed as discussed in the overview section. After each epoch, the error graph in the lower section of the network window will be updated with information on the current training and validation sample RMS.

The status of training after 36 epochs on the current model can be seen in Figure 17. The two lines represent the path of RMS for the training and validation sample as training has

proceeded[7]. In the upper right of the error graph, is a legend which shows the RMS for both samples after the latest training epoch. This view shows the network early in the training process. However, even during training the views and analyses discussed later can be performed on the current version of the network.

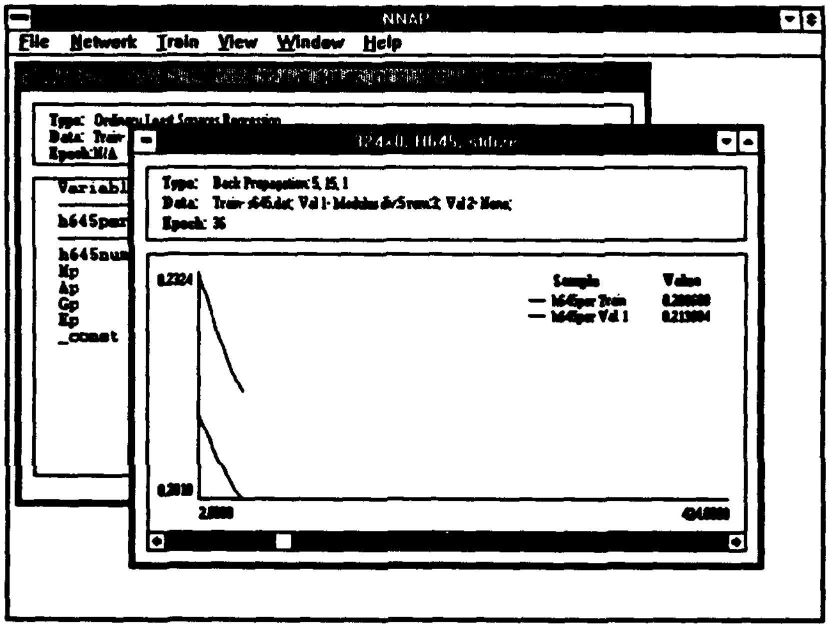


Figure 17.
Early training error paths for the training and validation samples.

The amount of time required for a training epoch depends on the number of input and output variables, the complexity of the network (number of connections between neurons), and the size of the data set. It is not uncommon for small simple problems to require 20 or 30 minutes of training. Large, complex problems may require over 24 hours. The current problem required about 30 minutes of training on a 33 mhz 80386 machine with a math co-processor before it was apparent that further training would be of no benefit.

Using Autoscale. Often the path of the training RMS will take the values off the bottom (or even the top) of the error graph window. To re-scale the graph to fit with the window the auto-scale, the **Autoscale** option should be selected from the **Train** menu or the pop-up menu.

Changing Graphed Variables. In our example with a single output, the current error graph is completely sufficient. If we had a second validation sample, its RMS would appear as a third line on the graph. However, if the model has several outputs, a graph of all training and validation sample RMS paths would be very cluttered. Using the **Error Variables** option under

[7]The upper line is the validation sample RMS while the lower is the training. These lines are different colors on a standard monitor.

the **Train** menu, the user can select which training and validation sample errors to graph. Figure 18 shows the SNNAP screen with this dialog box invoked.

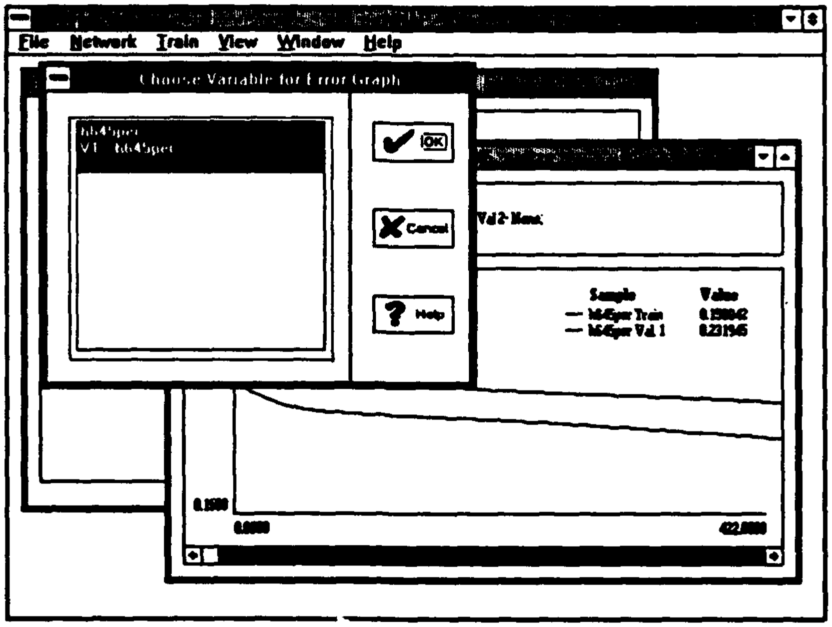


Figure 18.
Choosing what is shown on the network error graph.

Scrolling the Error Graph. When training proceeds for hundreds or thousands of epochs, the number of epochs will exceed the size of the error graph screen. In this case the graph can be scrolled using the scroll bars seen at the bottom of Figure 17. These scroll bars can also be used to locate training epochs of particular interest (such as validation sample minimums). It is also common to scroll the first few epochs of training off the screen before using **Autoscale.** The first few epoch typically have very high RMS which makes the rest of the training path difficult to see.

Getting an Overview of the Training Path. Often the training path can be difficult to visualize when hundreds or thousands of epochs have passed. The **Scale to Fit** option under the **Train** menu compresses the entire training path so that it fully appears in error graph window. The entire training path for the task performance problem can be seen in Figure 19. The minimum validation sample RMS can be clearly seen at about 600 epochs. At 1122 epochs, network training was manually stopped.

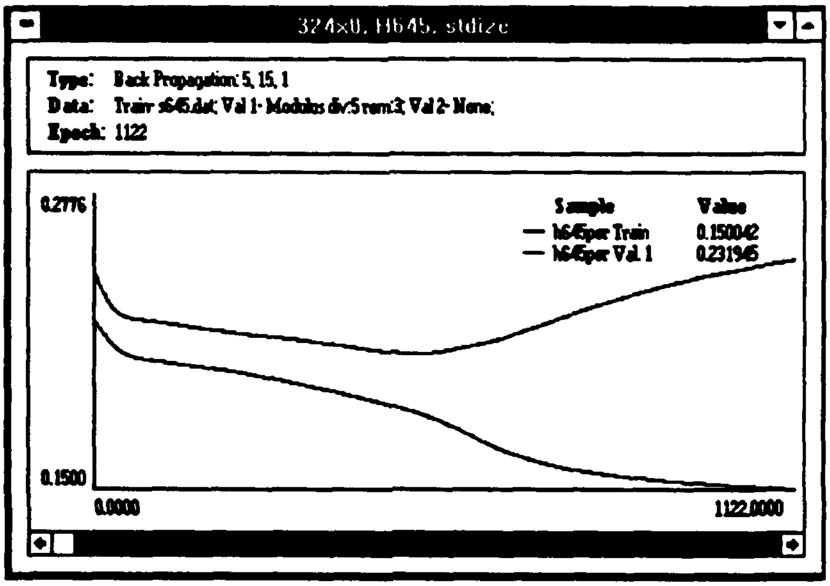


Figure 19.
The complete network training path.

Restoring Networks from Save Points. It is clear from looking at Figure 19 that the network which performs best on the validation sample is not at the end of training. To use the model with minimum validation sample RMS, the **Restore** option from the **Networks** window is used. This brings up the **Restore Weights** dialog box shown in Figure 20. As can be seen, the model with the best validation sample performance is selected; and this model will be used in all further analyses.

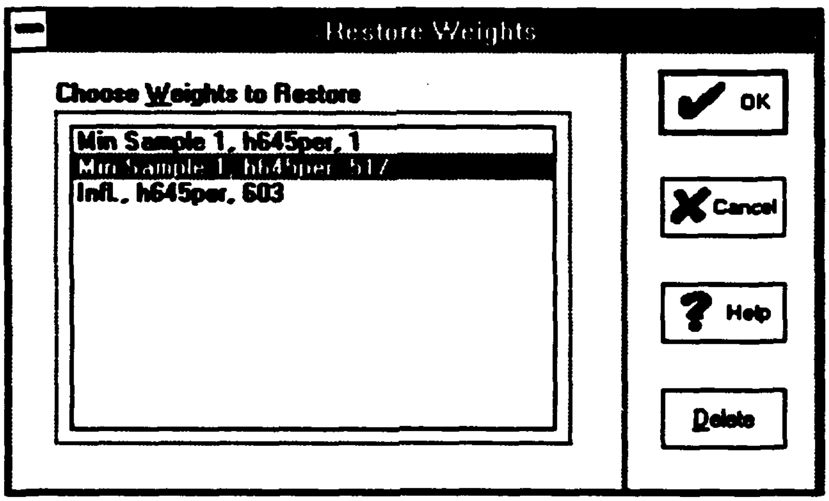


Figure 20.
Restoring network weights saved during training.

Comparing Model Performance

The first analysis we will perform involves comparing the training and validation sample performance of the OLS and back propagation (BP) network models. By activating the window for a model and selecting the **Statistics** option from the **Networks** menu, a set of estimation and validation sample performance statistics is produced for a model. This has been done for both the OLS and BP models and the section of the screen containing the results is shown in Figure 21. The OLS statistics are in the left window, with the BP model statistics in the right window.

Outside of the means and standard deviations (which are informative but do not compare performance), all of the statistics are derived from or related to the sum of squared prediction errors. Each of the RMS, TIC, R-squared, Janus Quotient, and Correlation are different scaled measures of the error. The Janus Quotient and TIC represent perfect prediction with 0 and larger values represent worse performance (TIC limited by infinity, the Janus Quotient by 1). The R-squared and actual/predicted correlation represent perfect models with 1. Actually, a Janus Quotient of 1 represents a model which performs no better than the mean of the actual output variable. If the model performs worse than the mean, Janus Quotient scores above 1 are possible. This same result holds for R-squared values below 0. These can be interpreted as models which perform worse than the actual mean of the output variable.

Statistic	Training	Validation 1
RMS	0.1988	0.2133
Actual Mean	0.8902	0.8853
Network Mean	0.8877	0.8662
Actual Std. Dev.	0.2105	0.2279
Network Std. Dev.	0.0691	0.0761
TIC	0.1557	0.1691
TICB	0.0002	0.0081
TICV	0.5060	0.5060
TICC	0.4938	0.4860
R squared	0.1081	0.1237
Janus Quotient	0.9444	0.9361
Correlation	0.3289	0.3627

324x0, H645, stdize : h645per

Statistic	Training	Validation 1
RMS	0.1763	0.1985
Actual Mean	0.8902	0.8853
Network Mean	0.8887	0.8569
Actual Std. Dev.	0.2105	0.2279
Network Std. Dev.	0.1069	0.1536
TIC	0.1378	0.1572
TICB	0.0001	0.0206
TICV	0.3451	0.1400
TICC	0.6548	0.8394
R squared	0.2982	0.2411
Janus Quotient	0.8378	0.8712
Correlation	0.5475	0.5274

Figure 21.
Comparing in- and out-of-sample performance statistics for OLS (left table) and back propagation (right table) models of airman performance.

As can be seen in the figure, the BP network fits the actual task performance measure better both in the training and validation samples. The differences can be seen most plainly in the R-squared and the correlation coefficient where the scale of these measures improves their

resolution in the error range of these models. It is interesting to note that the 0.3627 correlation for the OLS model on the 25 validation sample observations represents an insignificant correlation at the 5% level. Alternately, the .5274 validation sample correlation for the BP model is significant at the 5% level. Comparing the actual and network standard deviations, it can be seen that the OLS model shows much less variability in its predictions than exist in the actual data. While still considerably smaller than the actual standard deviation in the H645per variable, the network produces considerably more variation in its response than the OLS model. The importance of this can be seen by examining the TICV or variance component of the TIC. For the OLS model, about 50% of the prediction error, as measured by the TIC can be attributed to lack of variation in the OLS predictions. Alternately, on 35% training sample and 14% validation sample TIC error is attributed to lack of variation in the BP model.

Viewing the Response Surface

This section will introduce the view facilities available in SNNAP and demonstrate the response features which allowed the BP model to perform better in- and out-of-sample. In all cases, we will be using the two models just developed — the OLS model and the BP model.

Basic Views

2D Graphs. The tour of the visualization tools will begin with some simple 2-dimensional graphs. All views are initiated by selecting the View option from the View menu on the main menu bar (or from the pop-up menu). The dialog box for selecting a view will then be invoked as seen in Figure 22.

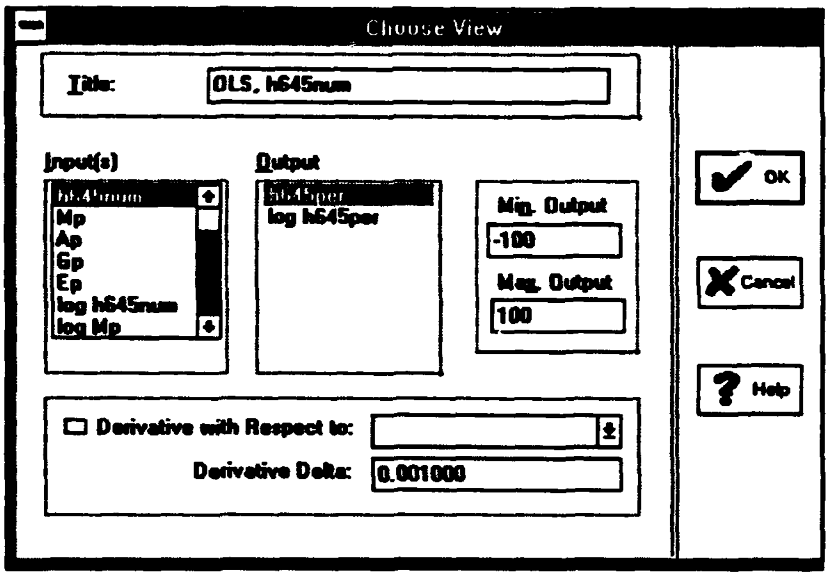


Figure 22.
Selecting a view of a network's response surface.

At the top of the box is a Title option which we have used to label the graph as coming from the OLS model. The OLS model will be used for this view because it was the current

window when the **View** option was present. Clicking on the BP network window would make that window current and allow **View** operations on the BP network. All operations operate on the active window.

The other options on the **Choose View** dialog box include selecting the input variables (select 1 or 2) and the output variable (select 1). As can be seen, the log transformations of the input and output variables are also available for graphing[8]. The section at the bottom of the box allows derivatives to be viewed and will be discussed later. For the current view, task performance (h645per) is selected against the number of times the incumbent had performed the task (h645num). This same selection process was performed for the BP model with the result of the two views shown in Figure 23. This Figure represents a section of the SNNAP screen after the two views were selected.

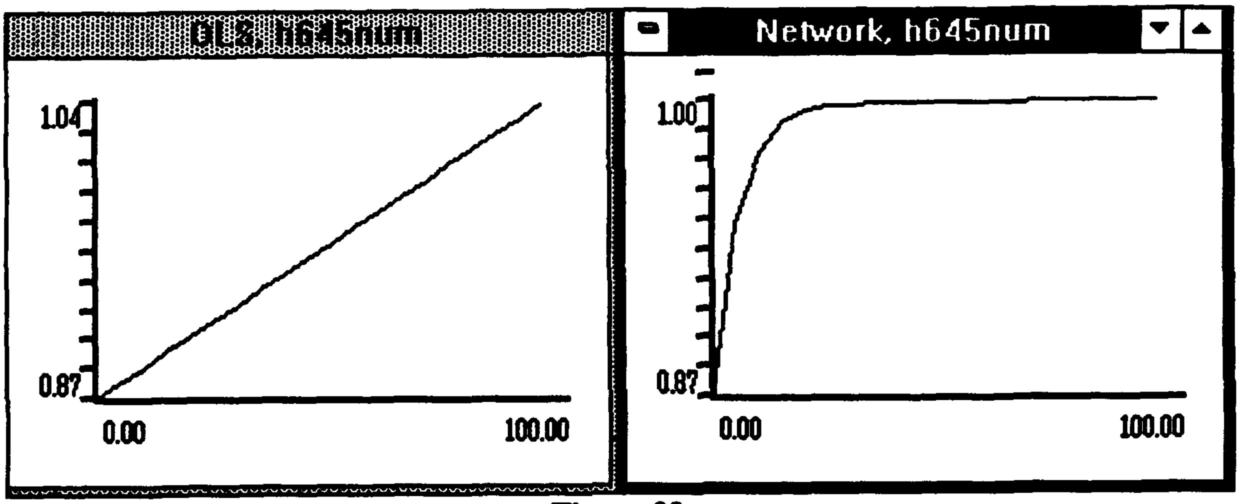


Figure 23.
The response of airmen performance to different levels of task experience. OLS model on the left, back propagation model on the right.

The two models clearly have a different opinion of the impact of task experience on task performance. While both models agree that the proportion of steps correctly completed is 0.87 for those with no task experience and about 1.00 for those with 100 repetitions performing the task, they differ radically in how the 100% performance is obtained. The network model postulates that proficiency on the task improves dramatically early in the experience path with complete proficiency obtained with fewer than 20 repetitions. Alternately, the OLS model, restricted by its linear form, postulates a steady improvement over the entire experience path. It should be noted that the form suggested by the network is not well approximated by simple transformations such as logs. It is most similar to a functional form requiring nonlinear estimation techniques and which is notoriously unstable to estimate.

---

[8]Note that variables with any negative or 0 values should not use the log transformations (eg. h645num where many job incumbents had never performed the task).

When looking at Figure 23, one should keep in mind that the graphs shown are merely a 2-dimensional slice out of a 6-dimensional response surface. For the OLS model, this point is irrelevant. The slope of the line shown will be the same regardless of the value of the other 4 variables (Mp, Ap, Gp, and Ep). As the other 4 variables change; the level, or intercept, of the line will of course vary according to the positive or negative coefficients on the other 4 variables. The interpretation of the graph produced by the BP network is radically different. The trained network model may contain features which cause not just the level, but also the impact of h645num to change as the other variables change. For example, the shape of the network curve in Figure 23 may be different for high aptitude airmen and low aptitude airmen.

3D Graphs. One way of directly visualizing the interactions just discussed is to examine 3-dimensional slices of the models response surface. To do this, the **View** option is again chosen for both models. However, this time both the h645num and Mp inputs are chosen (Mp was chosen because it had the largest coefficient in the OLS regression). The results of these two views are shown in Figure 24.

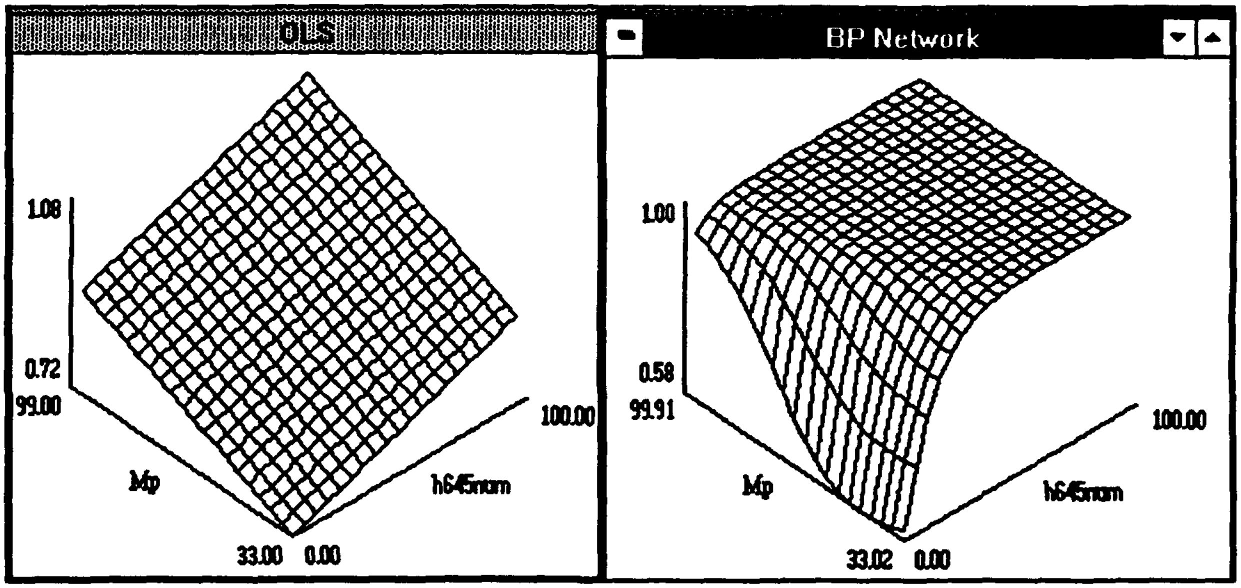


Figure 24.
The response of airmen performance to a range of levels of task experience (h645num) and mechanical aptitude (Mp). OLS model on the left, back propagation model on the right.

The graph of the OLS model is the expected plane in 3-D space. However, the BP network model shows a much more interesting structure. Those with very high mechanical percentile scores require almost no task experience to perform the "Calibrates Distortion Analyzers" task perfectly. Those with very low mechanical aptitude require many repetitions to achieve perfect performance (this is a task with a very high performance rating across individuals). It can also be seen that performance improves dramatically with very few repetitions for those with low and middle Mp percentile scores. While all Mp percentile groups eventually produce maximum performance (as measured here), the amount of task training

required to attain this performance is directly related to aptitude as measured by Mp. The BP network also shows a much wider response over the input values (.58 to 1.00) than the OLS model (.72 to 1.08). This is consistent with the higher variation seen in the BP model statistics.

For comparison, the response of performance, as measured by the network, to various levels of task experience (h645num) and Ep is shown in Figure 25. The range of performance is much lower in this view (.85 to 1.0 vs. .58 to 1.0 for Mp). In addition, high scores on Ep are not as indicative of early job performance as high Mp scores. This latter result is consistent with the smaller and less significant OLS coefficient values seen in Figure 12. Over the response surface seen in Figure 25, the impact of Ep on job performance is very small and linear. The effect of task repetitions continues to show the characteristic structure seen earlier in Figures 23 and 24.

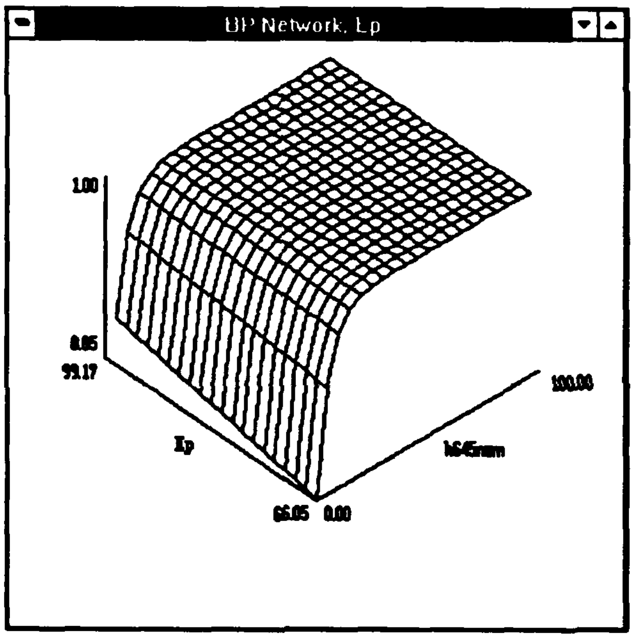


Figure 25.
The response of airmen performance to levels of task experience and electronic aptitude.

Toggling Tables and Graphs

The graphical views provide an intuitive approach in examining network response surfaces. However, in many cases it is important to quantify the relationships developed by the network. By selecting the **Table** option from the **View** menu or the pop-up menu, a tabular view of the graph can be produced. This option actual toggles to a tabular view of the intersection points on the graphical view (select the **Graph** option toggles back to a graphical view).

Figure 26 shows the results of toggling the two graphs from Figure 23 to the tabular view. We see the modeled level of performance for various numbers of repetitions. As can be seen, both networks model very similar levels of performance for those with no task experience (0.872 for OLS and 0.874 for BP)[9]. However, they model decidedly different pathways to full proficiency. At just over 5 repetitions, the network model projects almost 95% of steps completed correctly. The OLS model projects over 42 repetitions required to reach this same performance.

OLS, h645num		Network, h645num	
h645num	h645per	h645num	h645per
0.000	0.872	0.000	0.874
5.263	0.881	5.263	0.947
10.526	0.890	10.526	0.978
15.789	0.899	15.789	0.991
21.053	0.908	21.053	0.996
26.316	0.916	26.316	0.998
31.579	0.925	31.579	0.999
36.842	0.934	36.842	0.999
42.105	0.943	42.105	0.999
47.368	0.951	47.368	0.999
52.632	0.960	52.632	1.000
57.895	0.969	57.895	1.000
63.158	0.978	63.158	1.000
68.421	0.987	68.421	1.000
73.684	0.995	73.684	1.000
78.947	1.004	78.947	1.000
84.211	1.013	84.211	1.000
89.474	1.022	89.474	1.000
94.737	1.031	94.737	1.000

Figure 26.
Tabular view of task performance over a range of task experience levels. OLS model on the left, back propagation model on the right.

Getting a Different Perspective

SNNAP offers many options for helping to interpret and analyze the three dimensional graphical views of network response. Selecting the **Options** item from the **View** menu produces the **Shading Options** dialog box, shown superimposed on the graph in Figure 27. This graph again shows the response of performance to various levels of Ep and task experience.

As seen in the figure, the **Shade According to Orientation** option has been chosen. This causes the surface to appear as though it were lit by a light source directly overhead. In this case, the brightness of the surface areas is a direct representation of its slope. Those areas

[9]Remember that this evaluation is for those persons with mean percentiles on all of the selector AIs (Mp, Ap, Gp, and Ep).

which are very dark, are regions where one or both of the input variables have a large effect on the output variable (task performance). Bright areas, are regions where neither variable has a large impact on the output variable and the response surface is flat. The shade of the surface is directly proportional to the total derivative with respect to both inputs.

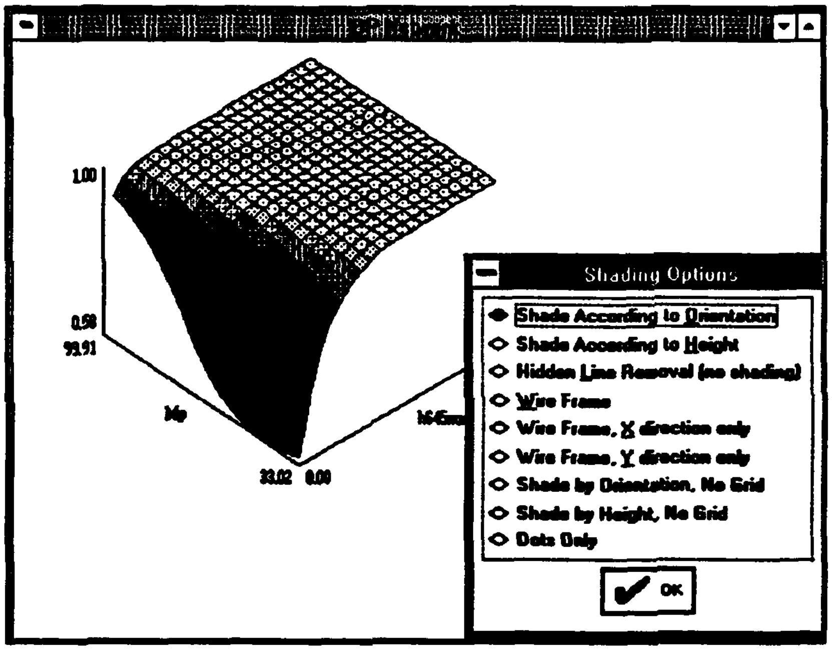


Figure 27.
The effect of shading a graphic view according to the orientation (or slope) of the surface.

A different perspective can be obtained by selecting the **Wire Frame, Y direction only** option. With this option, only those lines which connect the variable in the Y (task experience) dimension are drawn. Each line now represents a specific mechanical percentile score. In effect, several of the 2-D graphs shown in the left half of Figure 23 have been superimposed on the same graph. The only difference between each line is the Mp score.

This graph makes very apparent, the different task experience-performance profiles of airmen with different Mp scores. Those with lower scores have heavily curved lines which begin at just under 60% of steps correctly completed and rise rapidly to 100% of steps completed. Those airmen with high Mp scores begin their jobs with nearly complete proficiency.

The other options provide additional ways of modifying the graphical view. In particular, the **Shade According to Height** options colors each area of the surface according to its "height" or Z value. Areas with high proficiency are shown in a different color for those with low proficiency. Up to five color degradations can be used.

SNNAP also provides facilities for rotating three dimensional views. While the surface of the graphs presented thus far have been apparent from the default perspective, many times

significant features can be hidden from some perspectives of a surface. Selecting the **Rotate** option from the **View** menu invokes the **Rotate** dialog box seen in Figure 29. Clicking on the arrow buttons rotates the graph in the direction shown or specific orientation can be directly typed into the dialog box. The surface displayed in Figure 29 is a rotation of the surface in the right side graph of Figure 29 (and Figures 27 and 28). This different perspective adds little insight to the current graph, but does clearly demonstrate the rapid path to full proficiency and distinct differences across Mp score for those with little task experience.

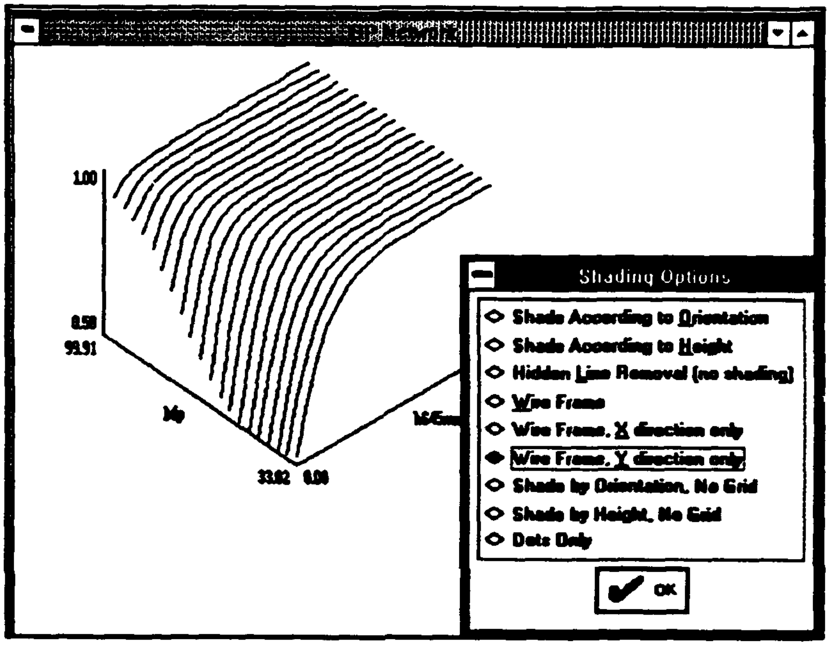


Figure 28.
The effect of connecting the wire frame in a single dimension for graphic views.

It is possible to combine the **Shade According to Height** option with rotation to produce a simple contour plot of the surface. When rotated such that the user is looking straight down the Z-axis of the graph, the colors represent the height for any point in X-Y space. This provides a block representation of a contour plot. The cut-off points for the colors used can also be set by the user. If the output were a binary decision, say reenlist vs. separate, the cut-off point could be set to 0.5 and the contour plot would then show the decision boundary between reenlist and separate decisions along any two input variables.

SNNAP also provides a **Scale** option for reducing the size of the displayed surface. It is available under the **View** menu or the current pop-up menu. The size of the view window can be adjusted using the standard MS Windows method of "grabbing" the lower right corner of the window. This will change the size of the window itself, but **Scale** must be used to reduce or enlarge the image of the surface (or line for 2-D graphs).

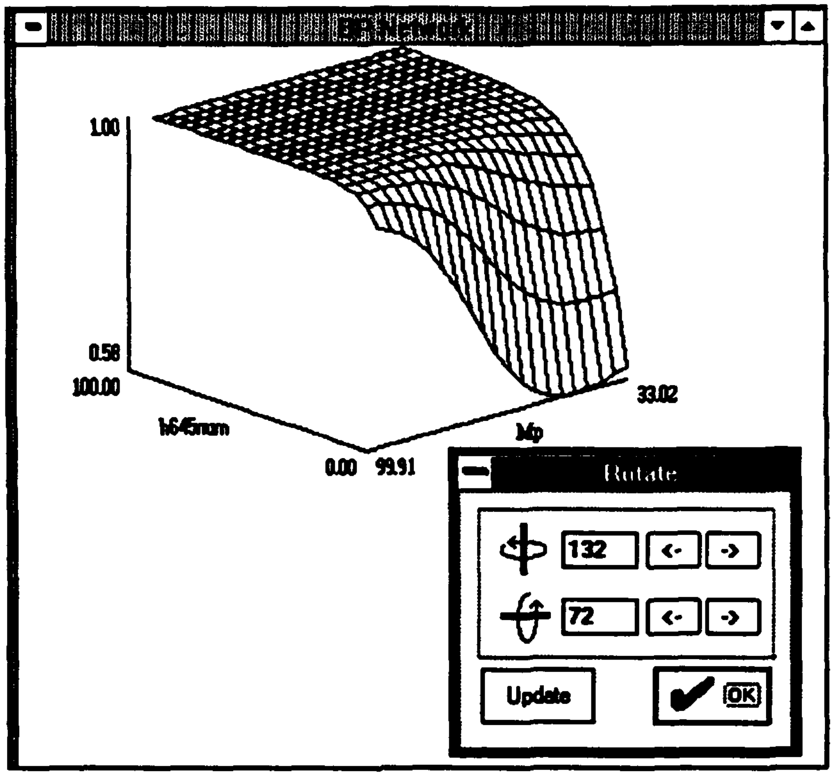


Figure 29.
Rotating a graphic view to change perspectives.

Changing the Area Viewed

As mentioned earlier, the 3-dimensional views are actually "slices" from a 6-dimensional space in which the current model operates. Up to this point, all of the views have assumed that all other variables (those not graphed) are taken to be fixed at their mean values over the training sample. It is of great interest to see if the same response holds for different levels of the other model inputs. The ability to change these default values is available under the **Defaults** option of the **Network** menu from the main menu bar. We saw the use of this option earlier (Figure 11) in the context of examining the model's variables.

Figure 30 shows the dialog box which allows the default value for any input variable to be changed. The default value is the value that will be used as input to the model when that variable is not one of the variables being analyzed in a view. As can be seen in the figure, the default value for Ep (in the **Values**) box has been set to 99. Originally, this default was set to the mean Ep value of 84.800285 seen in the upper box. A different variable can be selected by clicking on its name in the **Variable** box on the left. For this example the default values of Ap, Gp, and Ep were set to 99. This represents a person who ranks extremely high on all three of these selector AIs.

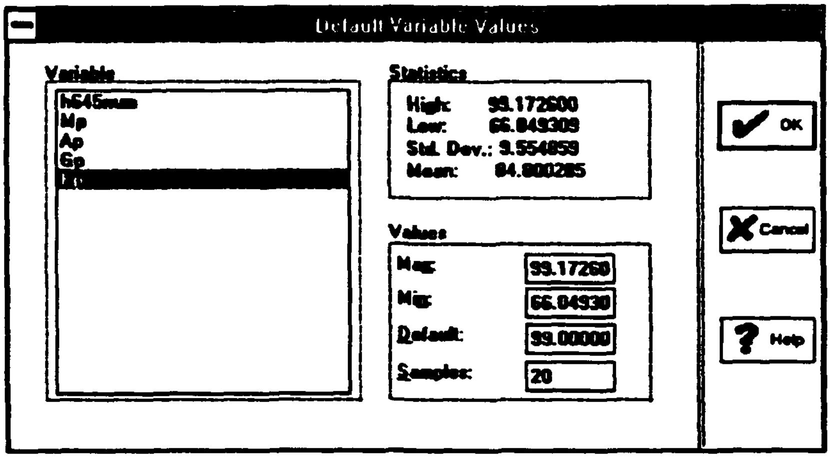


Figure 30.
Changing the Default value of variables which are not directly in a view.

When we then repeat the per645 vs. Mp and h645num graph with which we have been working, the results can be seen in Figure 31. The original graph is reproduced on the right for comparison with the graph of very high Ap, Gp, and Ep airmen on the left. In this case, the improvement due the high scores in all other areas has a minimal effect. The network models those with average Mp scores to improve somewhat faster to full proficiency, but those with very high or low Mp scores follow essentially the same training path regardless of the higher percentile scores on the other selector AIs.

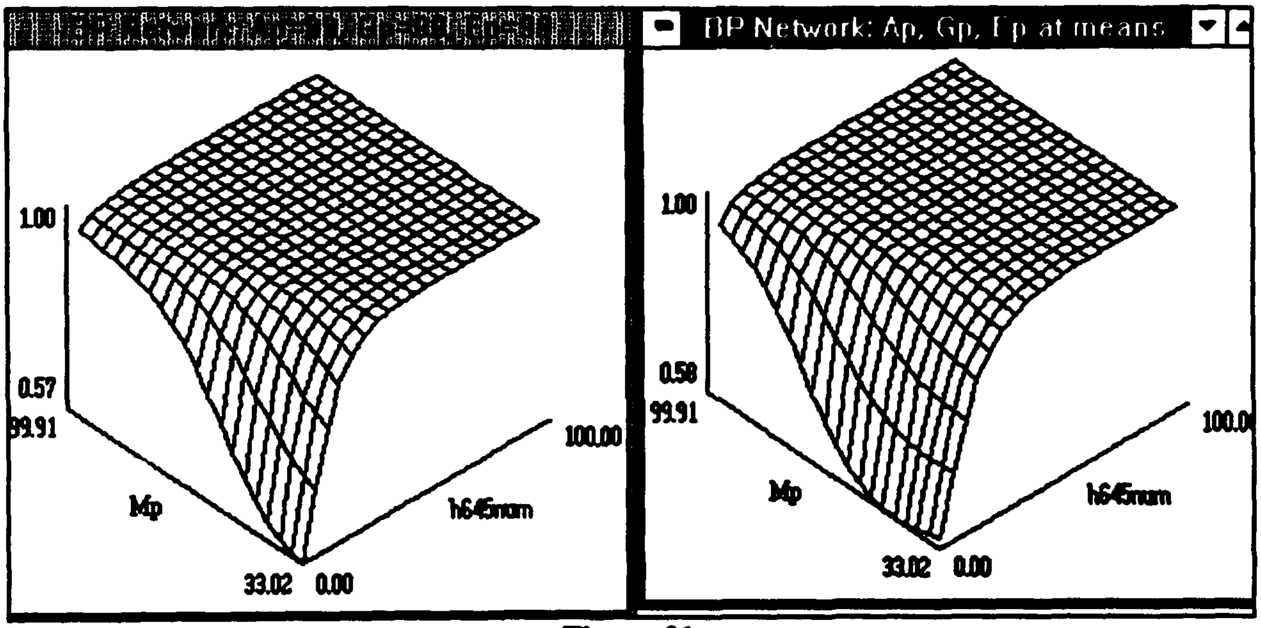


Figure 31.
The effect of task experience and mechanical aptitude for different levels of aptitude. High aptitudes on the left, typical aptitudes on the right.

In Figure 32, another use of the **Defaults** option is demonstrated. In this case, the **Min** and **Max** default values have been set to 40.0 and 100.0 respectively for the Mp variable. The **Max** and **Min** values control the range over which views will be computed. By default, these values are taken to be the maximum and minimum values found in the training data for the variable in question. However, more meaningful views can often be developed by limiting this range or extrapolating beyond the values found in the training data. In addition, the **Samples** value has been reset from the default of 20 to 7. This value controls the number of points between (and including) the minimum and maximum at which the network's response will be evaluated. In this case, we have chosen 7 samples between 40 and 100 which will produce evaluation points for every 10 additional Mp percentile points. In an additional step, the range of h645num was set to be 0 to 100 with 11 samples. Again this provides an even sampling every 10 task experience repetitions.

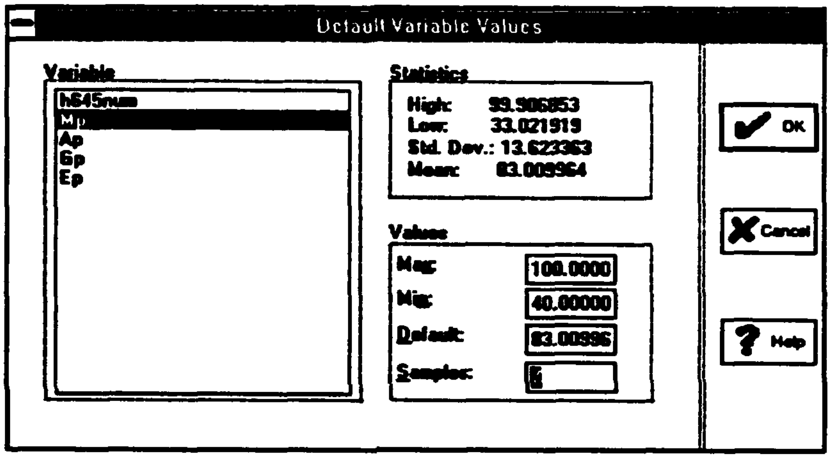
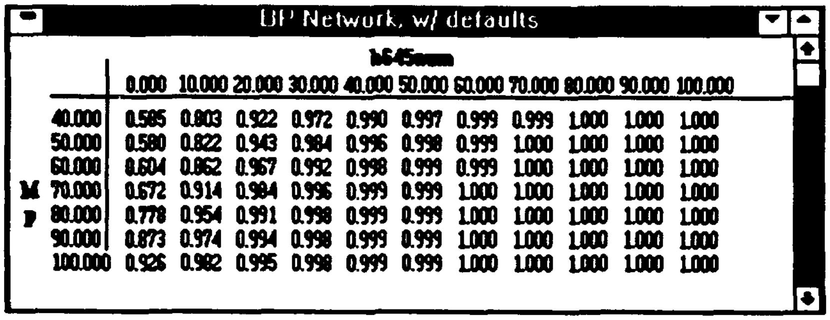


Figure 32.
Changing the range of values and number of samples used in creating a view.

The results of choosing these values and producing a tabular view of the response surface we have been analyzing can be seen in Figure 33. Each row represents an expected experience-proficiency path for airmen with different mechanical percentile scores. The changes made earlier to the **Max, Min,** and **Samples** on the **Default Variable Values** dialog box produced a table with both experience and aptitude broken down in regular sections. With this table, the user can more easily quantify the longer proficiency growth path seen in the graphs for those with low Mp scores.



BP Network, w/ defaults

Mp \ h645num	0.000	10.000	20.000	30.000	40.000	50.000	60.000	70.000	80.000	90.000	100.000
40.000	0.585	0.803	0.922	0.972	0.990	0.997	0.999	0.999	1.000	1.000	1.000
50.000	0.580	0.822	0.943	0.984	0.996	0.998	0.999	1.000	1.000	1.000	1.000
60.000	0.604	0.862	0.967	0.992	0.998	0.999	0.999	1.000	1.000	1.000	1.000
70.000	0.672	0.914	0.984	0.996	0.999	0.999	1.000	1.000	1.000	1.000	1.000
80.000	0.778	0.954	0.991	0.998	0.999	0.999	1.000	1.000	1.000	1.000	1.000
90.000	0.873	0.974	0.994	0.998	0.999	0.999	1.000	1.000	1.000	1.000	1.000
100.000	0.926	0.982	0.995	0.998	0.999	0.999	1.000	1.000	1.000	1.000	1.000

Figure 33.
Tabular view of the impact of levels of task experience and mechanical aptitude on task performance.

Using Automated Surface Scanning

Generating the Scan

SNNAP contains facilities to automatically search a response surface and note any distinctive features in the surface. As discussed earlier, it searches for linear, log-linear, linear-log, and log-log response over the entire area for which data is available. Any of these functional relations which remain constant over the range of the scan can be identified. Any other relation is flagged as unidentified. The scan also searches for interactions among inputs where the impact of one input on an output depends on the level of another input. A surface scan is performed by selecting the **Search** option from the **Network** menu. For the airmen performance network model, the window in Figure 34 will be generated.

The search process uses several tolerances to determine if relationships can be identified. Any of these tolerances can be changed by the user to adjust sensitivity of the search facility to slight deviations from zero impact, functional forms, and non-interacting effects. The **Zero Tolerance** setting seen in Figure 34 determines how much the input must affect the output for the scan to consider its effect as important. The tolerance is the proportion of the total range in the output which would be caused by a change in the input equivalent to its total range at the point where the impact is largest. For example, Mp ranges from 33 to 99 and h645num ranges from 0 to 100. If a change in Mp of 66 causes a change in h645num of more than plus or minus 10 (with a tolerance of 0.1) then the effect of Mp on h645num is determined to be greater than 0. When determining what impact a change in Mp of 66 will have, the most sensitive response at all of the training observations is used. The **Derivative Tolerance** is the tolerance ratio of the difference between the largest and smallest first derivatives and the largest first derivative. This tolerance determines whether an input is deduced to have a constant relation with an output (eg linear or log-log). The **2nd Derivative Tolerance** establishes the tolerance when testing for interactions among inputs. It is analogous to the **Zero Tolerance** except it tests whether any second derivative is non-zero. Once the tolerance (or tolerances) have been changed, selecting the **Calculate** button will generate a new search report.

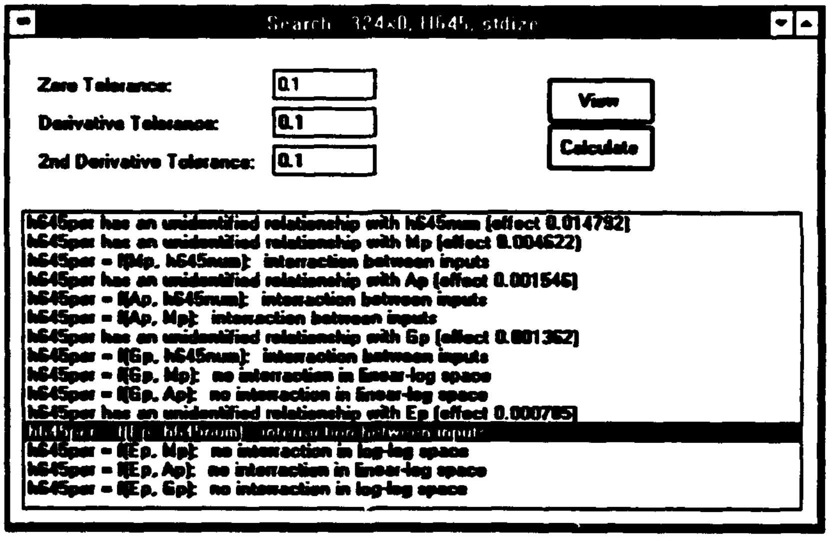


Figure 34.
Report window from searching a network's response surface.

The user can also control the range over which the scan is performed. Due to the problems inherent in searching high dimensional spaces, **Search** performs an analysis of the response surface in the neighborhood of each observation in the training data. Before being scanned, each observation is tested against the **Max** and **Min** currently set using the **Defaults** option. Normally, this test has no effect because the default **Max** and **Min** values correspond to the largest and smallest values found in the data. However, if the user wishes the search to be performed over a more restrictive range, these range values can be changed. This feature can be used to exclude outlying areas with sparse data coverage from the scan.

Interpreting the Scan

The results of the search appear in the list box in the window shown in Figure 34. This information can be used in several ways to gain a preliminary understanding of the model's response surface. Each input variable is tested singly against the output variable or variables to detect fixed relationships. Each line reporting one of these tests ends with an overall measure of the impact or effect of the input on the output. These lines and the overall effects can be identified in the figure by the **(effect x.xxxx)** suffix. This overall effect is simply the mean of the absolute first derivatives of the input on the output evaluated at each observation in the training data (or the limited set of data specified with the **Search** option). Larger values for the effect indicate a higher average impact of the input on the output. These effects are not scaled and will reflect the relative magnitudes of the input and output values. For example the effect value for h645num variable indicates that on average (over all training observations), a change of one task repetition causes about a 0.015 change in the proportion of correctly completed steps. This does not imply a direction for the impact. It does not even imply that the impact does not change signs over the response surface. It does give some indication of the typical size of the impact. Following each line indicating the one dimensional effect of an input, are a series

of lines indicating whether an interaction has been found among pairings with other inputs (these paired effects are symmetric and each pairing is found under only one initial input).

As can be seen in Figure 34, a fixed functional form cannot be identified for any of the inputs. Each input is designated by a line describing an "unidentified relationship" with the output variable. This implies that the effect of the task training and each of the aptitude variables varies over the response surface in a manner which cannot be captured by any of the fixed functional forms discussed earlier. It is possible to affect this "interpretation" by adjusting the **Derivative Tolerance**. As can be seen in the **Search** report, some of the input variables have interactions and some pairings of variables do not interact in linear or log-linear space. An example of an interaction has been shown extensively in the views of the relation between task experience, mechanical aptitude, and task performance.

Using Direct Links to Views

Each of the lines in the **Search** report can be used as a direct link to a view of the described relationship. By selecting one of the lines and the clicking on the **View** button, a view of the relationship (using the current **Defaults**) will be generated. This provides a quick way to visualize the described relationship. For example the results of selecting the line shown in black in Figure 35 is the view shown in the upper left corner of the figure. This view confirms the suggested non-interacting relationship (the lines in the surface plot may be nonlinear but are all relatively parallel). One should be aware that a single slice of the surface, such as that shown in Figure 35, can be misleading. While the surface may be flat for given values of the other variables, it may be nonlinear or of different slope for different values of the other input variables.

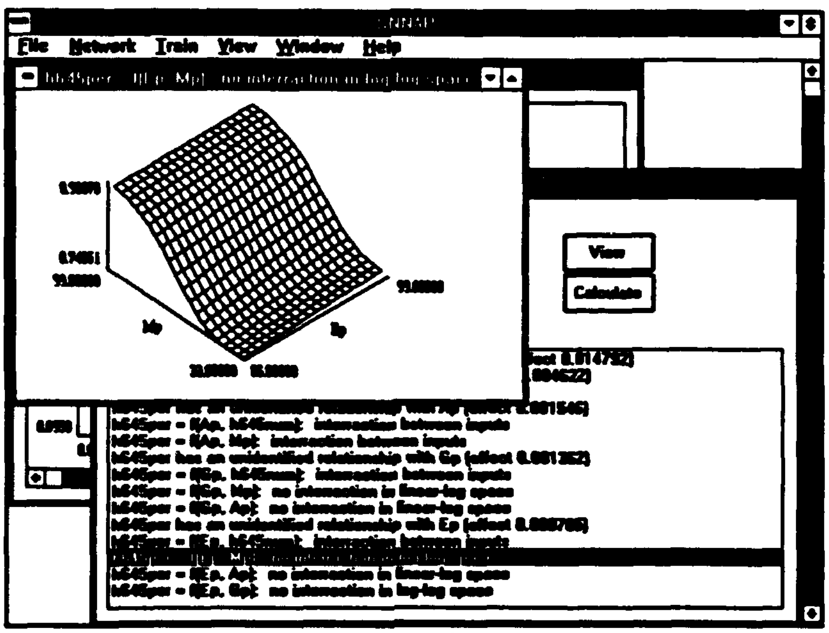


Figure 35.
Using direct links from the search report to views.

Keeping the Workspace Clean

Each of the major windows in SNNAP can be reduced to its icon using the standard windows method. By clicking on the reduce icon in the upper left corner of any window, it's icon representation will be placed at the bottom of the screen. Double clicking the icon will restore the window to its original size and position.

Figure 36 shows the icons for the primary SNNAP windows. Two views have been iconized and appear as the graph-like icons at the far left and third from the left in the figure. The second icon from the left represents the results of a network surface search. The icon at the far right represents a network window.

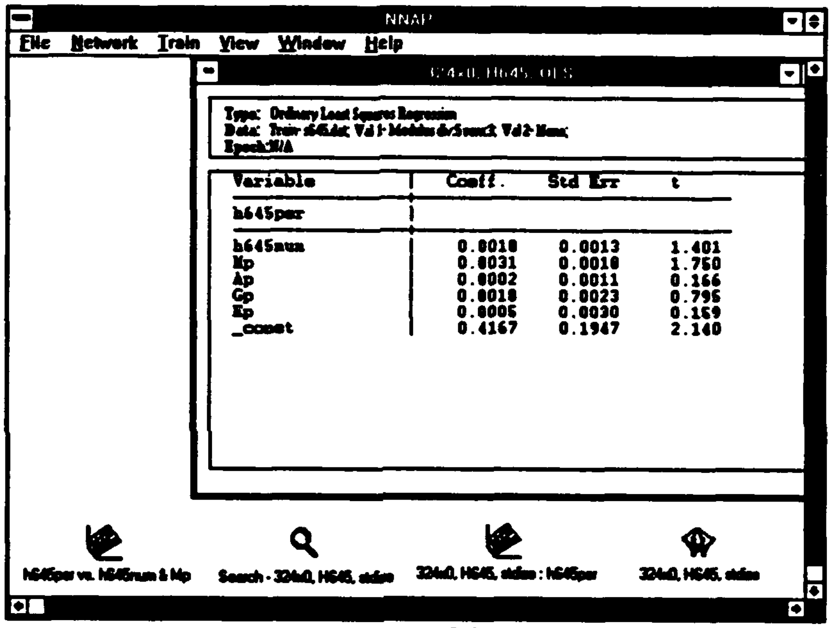


Figure 36.
The icons representing SNNAP windows.

Networks and search results can also be saved to disk using the **Save** or **Save as** options from the **File** menu or the current pop-up window. These windows can be restored to their complete state at the time of the save using the **Open** option. Networks can be saved, deleted, and later opened with no loss of information. Training can proceed from the point just prior to the save or any analysis carried out.

Views Revisited

The view facility provides several useful options which were not addressed earlier. These options can be used to analyze network behavior in more depth and provide different perspectives on model response. Logs and derivatives are the principal tools used to facilitate some aspects of network analysis.

Use and Interpretation of Logs

By taking the log of a variable in an analysis, the interpretation of its effect on an output changes. When an input is logorithmically transformed, a percentage change in the input produces the measured change in an output. When a transformed input forms a linear relation with an output, this implies a constant percentage change in the input is required to produce a constant absolute change in the output. If both the input and output are transformed, a linear relationship implies that a constant percentage change in the input produces a constant percentage change in the output. In many cases, this is an intuitively appealing interpretation (known in economics as constant elasticity models).

Log-log, log-linear, and linear-log effects can be analyzed by selecting the log variables shown in the **Choose View** dialog box for both the input and output variables. While it is entirely possible for the network to produce constant elasticity models in the current context, the fact that both the selector AI inputs and the proportion of steps correct output are already percentage measures makes elasticity interpretations unintuitive. The h645num variable cannot be transformed, as it contains many 0 values.

Views of Effects

Views of the effect of an input on the output as that input or other inputs change can provide further insight into a model's response. These views are obtained by producing graphs or tables of the derivative of an output with respect to an input for various levels of that input (or other inputs). Figure 37 demonstrates how these views are obtained in SNNAP. The check box at the bottom of the **Choose View** dialog box has been checked to indicate that derivatives rather than direct model output are to be viewed. As can be seen in the figure, the derivative is being taken with respect to the number of times the incumbent has performed the task (h645num). The results are interpreted as the impact on the proportion of tasks completed for an increase of one repetition of task experience.

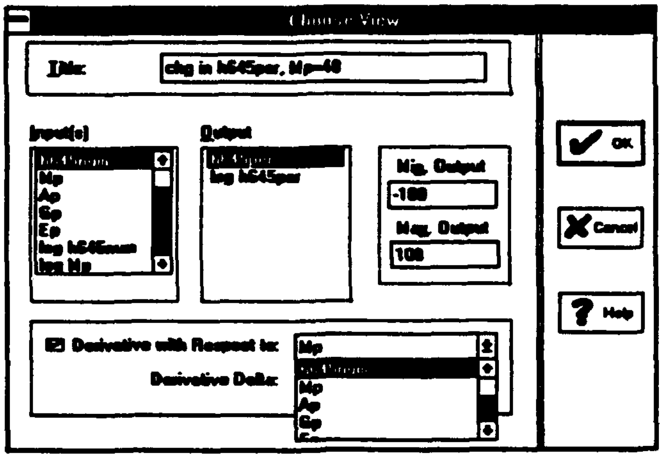
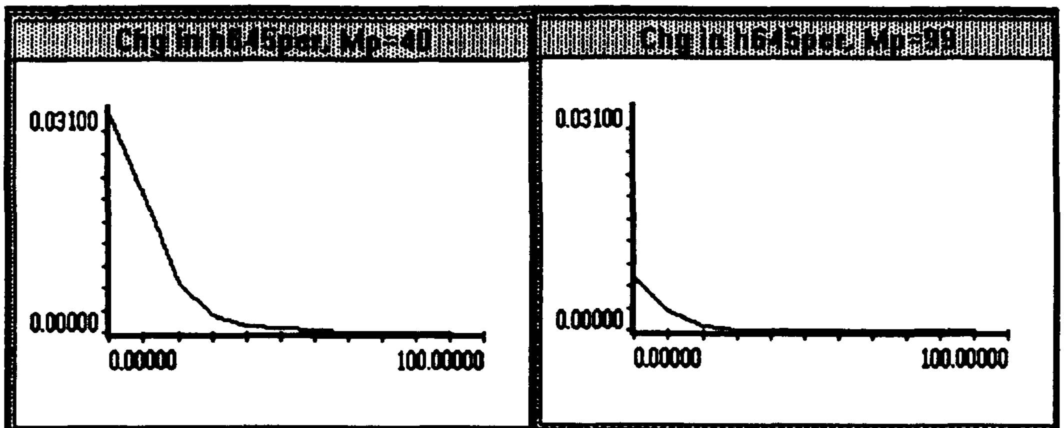


Figure 37.
Choosing an impact or derivative view.

We will use this facility to examine the difference between low and high mechanical AI scorers. The view selected in Figure 37 was generated for those with mechanical percentiles (Mp) of 40 and separately for those with mechanical percentiles of 99. The **Defaults** option was used to set the default value for mechanical percentile and to set the range and number of samples for the views. Figure 38 shows both the graphical and tabular forms of the views for these two percentile ratings. The **Range** view option has been used to put both graphs on the same scale for comparison.



Chg in h645per, Mp=40

h645num	h645per
0.000	0.031
10.000	0.020
20.000	0.007
30.000	0.003
40.000	0.001
50.000	0.001
60.000	0.000
70.000	0.000
80.000	0.000
90.000	0.000
100.000	0.000

Chg in h645per, Mp=99

h645num	h645per
0.000	0.008
10.000	0.003
20.000	0.001
30.000	0.000
40.000	0.000
50.000	0.000
60.000	0.000
70.000	0.000
80.000	0.000
90.000	0.000
100.000	0.000

Figure 38.
Changes in task performance given changes in task experience for low and high mechanical aptitude airmen.

The graphs show that those with high aptitude show much less improvement in performance for each additional task repetition. This is due largely to the very high initial performance for those with high aptitudes. Conversely, those with low mechanical experience and no task experience display almost a 3% increase in percent of steps completed for each additional task repetition. This rate of increase can be seen to decline to about 2% for those with 10 repetitions and just under 1% for those with 20 repetitions. By the time 40 task repetitions have been completed, very little further improvement is made. By this time, the

typical low mechanical aptitude airman has attained nearly 100% proficiency on the task (see Figure 28).

These views would make little sense for a linear regression model where the effect of any variable is constant for all values of that or any other model input. In this case, the views would all show a constant effect for all values of the input. With log-log or log-linear models, constant effects would be obtained when the appropriate inputs and/or output were designated as a log on the **Choose View** dialog box.

We can obtain a more complete view of the differential effects of task experience for different levels of experience and aptitude. Again, the derivative of task performance (h645per) with respect to task experience (h645num) is selected as the output variable. When task experience and mechanical aptitude (Mp) are selected as the input variable, the graph and table shown in Figure 39 are produced (these are actually two views of the same response surface).

Improvement in h645per per repetiti

h645num	Mp 40.000	50.000	60.000	70.000	80.000	90.000	100.000
0.000	0.028	0.031	0.034	0.036	0.032	0.021	0.011
10.000	0.019	0.020	0.020	0.015	0.009	0.005	0.003
20.000	0.008	0.008	0.005	0.003	0.001	0.001	0.001
30.000	0.003	0.002	0.001	0.001	0.000	0.000	0.000
40.000	0.001	0.001	0.000	0.000	0.000	0.000	0.000
50.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
60.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
70.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
80.000	0.009	0.000	0.000	0.000	0.000	0.000	0.000
90.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

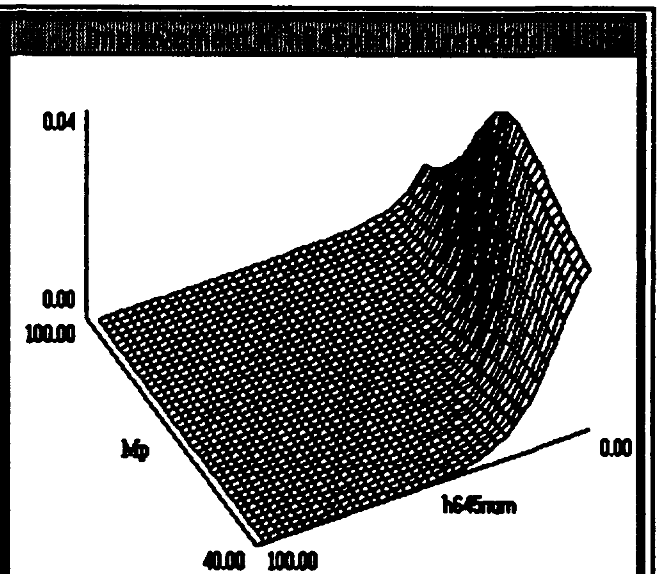


Figure 39.
Views of the change in task performance given changes in task experience for a range of mechanical aptitudes.

The graph shows the highest improvement in performance per task repetition for airmen with mid-level aptitudes[10]. Airmen with lower aptitudes demonstrate slower improvement at very low repetition levels but continue to improve at relatively high rates with more task experience. This relation can be quantified by examining the table. The highest rate of improvement is seen for airmen with no task experience and mechanical percentiles of 70 (3.6% improvement for each task repetition). Conversely, airmen with an Mp of 40 and no experience

[10]Note that the graph has been rotated to best reveal the surface. The lowest task experience levels are at the far right of the graph.

improve by 2.8% per repetition and those with an Mp of 100 improve by 1.1%. However, by the time task experience has reached 30 repetitions, those with an Mp of 40 continue to improve at the rate of 0.8% per repetition while those with an Mp of 70 improve by 0.3%. These observations on the rate of proficiency improvement as task experience increases help to quantify the relationship between aptitude, experience, and proficiency observed earlier in Figures 24, 27, and 28.

Analysis Summary

Many features of SNNAP have been demonstrated with this example probler... Performance of the network model was compared against a simple OLS model. With the tools available in the SNNAP environment, the structure of the trained network model was dissected and visualized.

In this example, the ability of the network model to project the performance of airmen not in the training sample was somewhat superior to the ability of the regression model. On the basis of this performance, an analysis of the network model's response surface revealed several interesting features.

While this analysis was limited to a single task in one AFS, many of the model's features would have significant policy implications if they were applied to selection and training. The Mp score appears to be a better indicator of task performance than the selector AI for the career field (Ep)[11]. All aptitude groups are capable of excellent task performance if task specific experience is sufficient. This hands-on training is not nearly as important for high Mp aptitude airmen as it is for those with lower Mp aptitude. In particular, hands-on training for the "Calibrates Distortion Analyzers" task is particularly effective for low and middle Mp aptitude airmen.

CONCLUSION

SNNAP is an environment for designing, training, and analyzing neural networks. It provides extensive facilities for visualizing and quantifying the relationships captured in a trained neural network. The performance of network models can be examined both in- and out-of-sample; and this performance can be compared to regression models within the SNNAP environment. SNNAP also implements automated facilities for suggesting network design and analyzing the surface of trained networks. It incorporates training heuristics to improve the ability of the network models to generalize to exemplars data outside the training data.

[11]This is supported by both the network and regression models.

As demonstrated in the example problem and prior research (Wiggins et al., 1992), neural networks can reveal complex nonlinear structure in models of many personnel decisions, behaviors, and systems. This structure often offers deeper insight into relationships and interactions among model determinants. As seen in the task performance example and prior research on reenlistment rates, the nonlinear features developed by networks often have significant implications for policy decisions. SNNAP offers the ability to easily search for and illustrate these nonlinear features in a neural network model. The software provides an environment to exploit the capabilities of neural networks in areas where model generalization and a deep understanding of the modeled relations is required.

REFERENCES

Cacoullos, T., "Estimation of a Multivariate Density", *Annals of the Institute of Statistical Mathematics (Tokyo)*, 18, 2, pp 179-89, 1966.

Duda, R, & P. Hart, *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York, 1973.

Durbin, R. & Rumelhart, D.E. (1989). Product units: a computationally powerful and biologically plausible extension to backpropagation networks. *Neural Computation*, 1, 1, 133-142.

Elman, J.L. (1990). Finding structure in time. *Cognitive Science*, 14, 179-211.

Hedge, J.W. (1984). *The methodology of walk-through performance testing*. Paper presented at the annual meeting of the American Psychological Association, Toronto.

Hedge, J.W., & Teachout, M.S. (1986). *Job performance measurement: a systematic program of research and development* (AFHRL-TP-86-37, AD-A174 175). Brooks AFB, TX: Training Systems Division, Air Force Human Resources Laboratory.

Hornik, K., Stinchcomebe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5), 359-366.

Kohonen, T. (1984). *Self-organization and associative memory* (3rd ed.). New York: Springer-Verlag.

Kohonen, T., Barna, G., & Chrisley, R. (1988). Statistical pattern recognition with neural networks. *IEEE International Conference on Neural Networks, San Diego, California, July, 1988*, 1, I - 61-88.

Lance C.E., Hedge, J.W., & Alley, W.E. (1987). *Ability, experience, and task difficulty predictors of task performance* (AFHRL-TP-87-14). Brooks AFB, TX: Manpower and Personnel Division, Air Force Human Resources Laboratory.

Morgan, N, & Bourlard, H. (1990). Generalization and parameter estimation in feedforward nets: some experiments. *Neural Information Processing Systems 2*, Touretzky, D.S. (ed.), San Mateo, CA: Morgan Kaufmann Publishers, 630-637.

Parzen, E., "On Estimation of a Probability Density Function and Mode", *Annals of Mathematical Statistics*, vol. 33, pp. 1065-76, 1962.

Rumelhart, D.E. (1990). Brain style computation: neural networks and connectionist AI (oral presentation). Las Vegas TIMS/ORSA Joint National Meeting, May 7-9, 1990.

Rumelhart, D.E., Hinton, G.E., & Williams, R.J. (1986). Learning internal representations by error propagation, in *Parallel distributed processing: explorations in the microstructure of cognition*, D.E. Rumelhart & J.L. Mclelland (Eds.). Cambridge, MA: MIT Press, 213-362.

Specht, D.F. (1990). Probabilistic neural networks. *Neural Networks*, 3(1), 109-118.

Stone, B.M., Looper, L.T., & McGarrity, J.P. (1990). *Validation and reestimation of an air force reenlistment analysis model* (AFHRL-TP-90-42). Brooks AFB, TX: Manpower and Personnel Division, Air Force Human Resources Laboratory.

Vance, R.J., MacCallum, R.C., Coovert, M.D., & Hedge, J.W. (1989). Construct models of task performance. *Journal of Applied Psychology*, 74, 3, 447-455.

Weiss, S.M., & Kulikowski, C.A. (1991). *Computer systems that learn.* San Mateo, CA: Morgan Kaufmann.

Wiggins, V.L. (1990). *Neural network applications literature review.* Informal technical report for Air Forces Human Resources Laboratory, Brooks AFB, TX (Contract No. F41689-88-D-0251, Task 29). Bryan, TX: RRC, Inc.

Wiggins, V.L., Looper, L.T., & Engquist, S.E. (1991). *Neural networks and their application to air force personnel modeling* (AL-TR-1991-0031). Brooks AFB, TX: Human Resources Directorate, Manpower and Personnel Division, Armstrong Laboratory.

Wiggins, V.L., Engquist, S.E., & Looper, L.T. (1992). *Applying neural networks to Air Force personnel analysis* (AL-TR-1991-0118). Brooks AFB, TX: Human Resources Directorate, Manpower and Personnel Division, Armstrong Laboratory.

APPENDIX A: Layout of Format Files

As discussed in the main report, SNNAP requires format files to identify the contents of a data set. The format file tells SNNAP where to find variables in a file and what the variables should be called. A very simple format file structure is required by SNNAP. Each line in the format file describes one variable to be available in SNNAP and requires the following five fields:

VariableName VariableType StartingColumn VariableFieldLength ClassNames

Each of the fields in the format file is defined as follows:

VariableName: The name to be used to identify the variable in SNNAP.

VariableType: A single character code for the type of variable. Most variables are treated as floating points (ie. reals) internally. The other types available are:

f = floating point
b = binary (0 or 1)
c = categorical (integer category codes)

StartingColumn: The column in the file where the field containing the variable begins. That is, the character position in a record where the field begins. This column is used only for fixed format data and is ignored for free format or delimited data.

VariableFieldLength: The length of the field containing the variable in the data file records (the length in characters).

ClassNames: Used only for categorical data types. Each space separated token names a class which is designated by the integer in the data set. The first token provides a name for 0, the second for 1, the third for 2, and so on. For binary and floating point numbers this field is not used but must contain a string.

With fixed format data, the order of the lines in the format file is unimportant. Unnecessary fields or regions of data in the data file can be ignored by not including these regions in the defined variables. In fact, variables can share characters in a data line (eg. a full date in yymmdd format can be read in total with the yy (year) component read into a separate variable). For free format and delimited data, each field in the data set must have a name in the format file. In addition, the order of the names in the format file will assume to hold in the data file. It is possible in free format data for a single line in a data file to contain several

logical records. The end of a line has no meaning for this file format. The format file for the task performance problem examined in the report is reproduced below.

```
h645per    f    1     10    unused
h645tim    f    13    4     unused
h645stim   f    19    4     unused
h645num    f    25    10    unused
expmos     f    37    10    unused
Mp         f    49    4     unused
Ap         f    55    4     unused
Gp         f    61    4     unused
Ep         f    67    4     unused
afqt2p     f    73    4     unused
```

APPENDIX B: SNNAP Flowchart

SNNAP was developed in Borland C++ using object oriented design and implementation methods. It was designed to operate in the Microsoft Windows environment. This environment operates in a an event loop paradigm which places the user in control of the program's execution path. For these reasons, extensive flowcharting is inappropriate for SNNAP. The figure below provides an overview flowchart of SNNAP from a very high level.

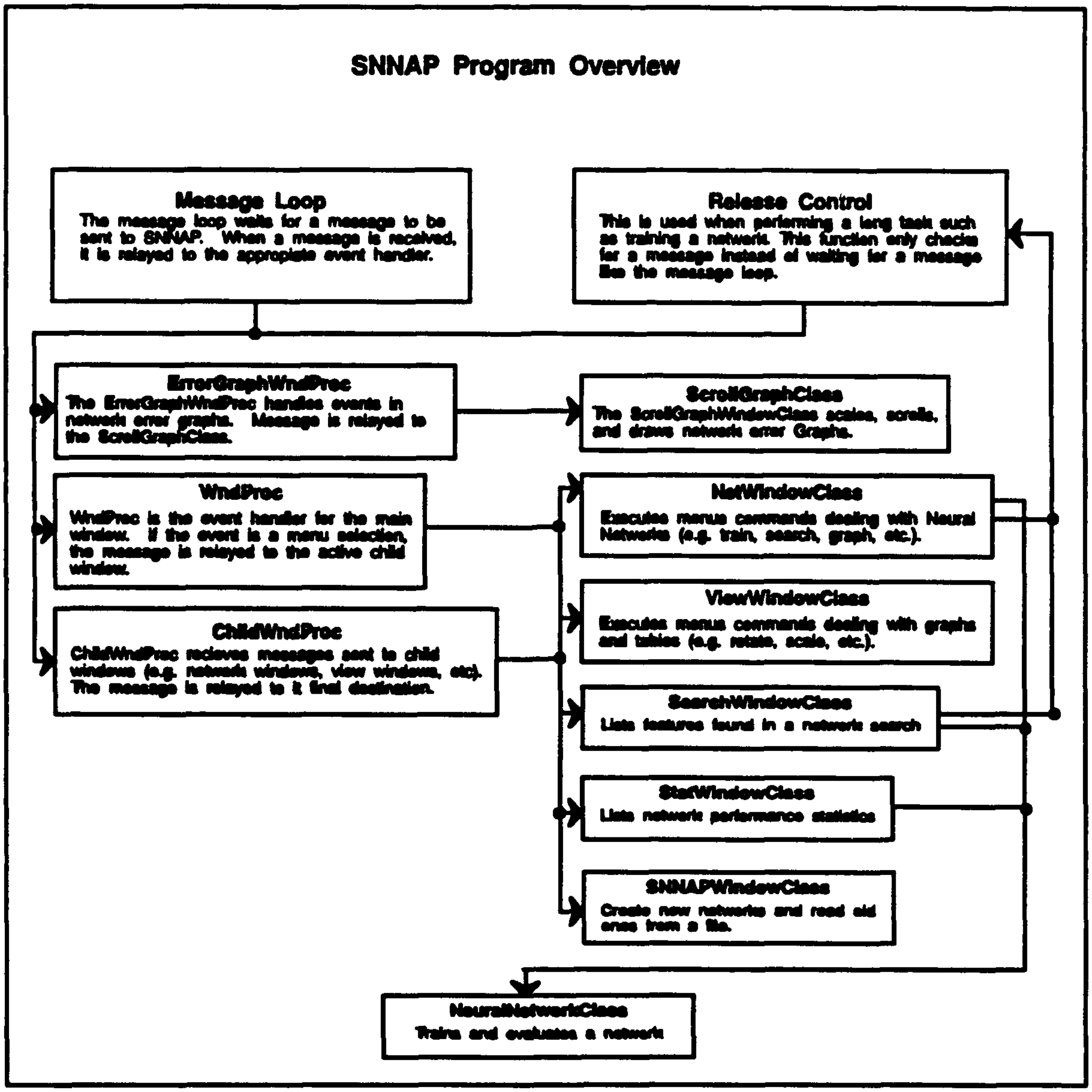


Figure B-1.
Overview flowchart for SNNAP.

APPENDIX C: 30 Steps in Task H645

In the airman performance example, the WTPT results for the "Calibrates Distortion Analyzers" task in the AFS 324X0 (Precision Measuring Equipment Specialists) were used to measure airman performance. In particular, the proportion of correctly performed of steps in the "Calibrates Distortion Analyzers" task were used as a measure of performance on the task. The WTPT recognized 30 steps for this task and the h645per measure used in the neural network and OLS models is the ratio of correctly performed to total steps. The steps used in the WTPT are listed below.

1 Select signal generator that meets specified range and accuracy.
2 Set test oscillator output to zero (minimum).
3 Connect standard output to test instrument input (properly terminated).
4 Set test instrument function range to set level.
5 Set test instrument frequency range to X1.
6 Set test instrument frequency dial to 10.
7 Set test instrument meter range to set level.
8 Set test instrument sensitivity to full CW.
9 Set test instrument sensitivity vernier to full CW.
10 Set test instrument mode to manual.
11 Set standard to 10HZ.
12 Set standard output control for O DB on test instrument meter.
13 Set test instrument function to distortion.
14 Adjust test instrument frequency dial and balance controls for null.
15 Set test instrument meter range to set level.
16 Set test instrument function to set level.
17 Set standard to 20HZ while adjusting standard output for O DB on test.
18 Set test instrument function to distortion.
19 Check that test instrument meter indicates between O and +1 DB.
20 Set test instrument function to set level.
21 Set test instrument frequency dial to 20.
22 Set standard to 20HZ.
23 Set standard output control for O DB on test instrument meter.
24 Set test instrument function to distortion.
25 Adjust test instrument frequency dial and balance controls for null.
26 Set test instrument meter range to set level.
27 Set test instrument function to set level.
28 Set standard to 40 HZ while adjusting standard output for O DB on.
29 Set test instrument function to distortion.
30 Check that test instrument meter indicates between -.6 AND +.6 DB.